· 高等学校计算机基础教育教材精选 ·

"国家级精品课程"配套教材

Web技术应用基础（第2版）

樊月华 主编
刘雪涛 刘洪发 编著

清华大学出版社

高等学校计算机基础教育教材精选

Web 技术应用基础
（第 2 版）

樊月华　主编
刘雪涛　刘洪发　编著

清华大学出版社
北京

内 容 简 介

本书以开发 Web 应用的工作过程为序，介绍了 Web 技术三个层面的应用。

全书共 3 篇 10 章，以一个与公司合作开发的实际案例——网上书店贯穿本书。第 1 篇"Web 技术基础"分为 3 章，主要内容是 Web 技术概述、Web 应用环境构建技术和网上书店的系统设计。第 2 篇"Web 客户端程序设计基础"分为 3 章，分别介绍了 HTML、CSS 和 JavaScript 技术。第 3 篇"JSP Web 数据库应用开发"分为 4 章，分别介绍了 JSP 运行机制与基本语法、JSP 内置对象、基于 JSP 的 Web 数据库应用开发和网上书店的实现。本书共 80 多个案例，以及一个实际案例——网上书店，它们的源代码可在清华大学出版社网站下载。

本书适合作为高等院校信息技术教材，也可以作为 Web 应用开发人员的培训教材和入门参考书。

图书在版编目（CIP）数据

Web 技术应用基础/樊月华主编；刘雪涛，刘洪发编著. —2 版. —北京：清华大学出版社，2009.1
（高等学校计算机基础教育教材精选）
ISBN 978-7-302-18840-7

Ⅰ. W… Ⅱ. ①樊… ②刘… ③刘… Ⅲ. 计算机网络－程序设计－高等学校－教材
Ⅳ. TP393.09

中国版本图书馆 CIP 数据核字(2008)第 169365 号

责任编辑：焦　虹　王冰飞
责任校对：李建庄
责任印制：

出版发行：清华大学出版社　　**地　　址**：北京清华大学学研大厦 A 座
http://www.tup.com.cn　　**邮　　编**：100084
社　总　机：010-62770175　　**邮　　购**：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质　量　反　馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 刷 者：
装 订 者：
经　　销：全国新华书店
开　　本：185×260　**印　张**：19.5　**字　数**：458 千字
版　　次：2009 年 1 月第 2 版　**印　次**：2009 年 1 月第 1 次印刷
印　　数：1～0000
定　　价：0.00 元

出版说明

在教育部关于高等学校计算机基础教育三层次方案的指导下，我国高等学校的计算机基础教育事业蓬勃发展。经过多年的教学改革与实践，全国很多学校在计算机基础教育这一领域中积累了大量宝贵的经验，取得了许多可喜的成果。

随着科教兴国战略的实施以及社会信息化进程的加快，目前我国的高等教育事业正面临着新的发展机遇，但同时也必须面对新的挑战。这些都对高等学校的计算机基础教育提出了更高的要求。为了适应教学改革的需要，进一步推动我国高等学校计算机基础教育事业的发展，我们在全国各高等学校精心挖掘和遴选了一批经过教学实践检验的优秀的教学成果，编辑出版了这套教材。教材的选题范围涵盖了计算机基础教育的三个层次，包括面向各高校开设的计算机必修课、选修课，以及与各类专业相结合的计算机课程。

为了保证出版质量，同时更好地适应教学需求，本套教材将采取开放的体系和滚动出版的方式(即成熟一本、出版一本，并保持不断更新)，坚持宁缺毋滥的原则，力求反映我国高等学校计算机基础教育的最新成果，使本套丛书无论在技术质量上还是文字质量上均成为真正的"精选"。

清华大学出版社一直致力于计算机教育用书的出版工作，在计算机基础教育领域出版了许多优秀的教材。本套教材的出版将进一步丰富和扩大我社在这一领域的选题范围、层次和深度，以适应高校计算机基础教育课程层次化、多样化的趋势，从而更好地满足各学校由于条件、师资和生源水平、专业领域等的差异而产生的不同需求。我们热切期望全国广大教师能够积极参与到本套丛书的编写工作中来，把自己的教学成果与全国的同行们分享；同时也欢迎广大读者对本套教材提出宝贵意见，以便我们改进工作，为读者提供更好的服务。

我们的电子邮件地址是：jiaoh@tup.tsinghua.edu.cn；联系人：焦虹。

清华大学出版社

前言

人类已进入信息社会,信息技术(IT)的应用渗透到各个领域。基于 Web 的应用系统正在成为信息系统的主流。Web 技术也是 IT 领域的一项关键技术,随着网上应用系统、企事业信息管理、电子商务和电子政务等需求的增加,使用 Web 方式进行信息处理和应用系统开发已经成为主流发展趋势。Web 技术是网上信息应用的基础,是信息管理、计算机等专业,甚至是 IT 类专业的一项主要技术基础,它也是从事信息事业的技术人员和管理者需要掌握的重要技能之一。

一个基于 Web 技术的应用开发需要三个层面的技术,本书根据这三个层面设计了三个台阶,读者每完成一部分内容的学习,即可进行 Web 应用开发的某一方面的工作。

(1) Web 应用环境构建技术。能力要求:掌握构建 Web 应用软硬件平台的基本技能。

(2) Web 客户端程序设计基础。能力要求:具有基本的 Web 编程能力。

(3) Web 数据库应用开发技术。能力要求:熟练掌握应用 JSP 技术完成数据库信息存储、管理与发布技术。

在 Web 应用环境构建部分,重点介绍了 JSP 运行环境的构建。Web 客户端编程技术是 Web 应用开发的基础,本书对这部分内容介绍以必需、够用为度,重点介绍了 HTML、CSS 和 JavaScript 技术的应用。如果读者具有某种程序设计语言的基础,这部分内容可以通过本书案例自学。基于 Web 技术的应用开发的核心技术是应用逻辑处理技术和数据库信息发布技术,这是本书讲解的重点。

本书是在作者完成多轮教学与多个项目的基础上写成的,全面地介绍了 Web 技术应用的基本概念与理论,详细讲解了市场主流成熟技术的应用。本书具有以下三个特点:

(1) 以应用为导向。本书以应用为导向,以 Web 应用开发过程较为全面地介绍了市场主流和成熟技术的应用。

(2) 将实际项目引入教学。本书的案例“网上书店”是一个实际项目,是作者与企业共同开发的,为便于教学,对部分内容作了适当简化。本书介绍了“网上书店”从设计到实现的全过程,读者按照系统使用说明,可将系统恢复,用以教学或参照开发新的应用。

(3) 以案例带动教学。本书以案例带动教学,强调在做中学,将实践与学习结合。本书共提供案例 80 多个,供读者学习使用。

书中每章后面有大量的习题、上机练习和实训课题,其目的是使学生掌握核心知识、概念和技术。在实训中还提供了一些综合应用的课题。

本书由樊月华、刘雪涛和刘洪发共同策划，第 4、5、6 和 10 章由刘雪涛编写，第 1、2 和 3 章由刘洪发编写，第 7、8、9 章由樊月华编写。在网上书店项目设计与开发过程中，得到多特教育责任有限公司的鼎力协助，在此一并感谢。

本书第 2 版更新了软件版本，对部分内容和例题进行了更新，保证了教材的先进性。并将内容集中于 JSP 技术，删减了 ASP. NET 部分内容，使内容更加紧凑。

感谢读者选择使用本书，欢迎对本书结构、内容提出批评和修改建议，本书不当之处敬请指正，我们将不胜感谢。

作 者

2008 年 11 月

目录

第 1 篇　Web 技术基础

第 3 篇　JSP Web 数据库应用开发

第 1 篇　Web 技术基础

本篇主要介绍与 Web 技术相关的基础知识与原理。通过第 1 篇的学习，读者将了解 Web 应用的基本知识与原理，通过一个实际案例（网上书店，此案例将贯穿于全书）了解基于 Web 方式的应用系统开发的全过程，并掌握 Web 应用环境的构建技术。本篇主要内容包括：

第 1 章　Web 技术概述

第 2 章　Web 应用环境构建技术

第 3 章　基于 Web 方式的信息系统开发案例——网上书店

第 1 章 Web 技术概述

本章主要介绍 Web 技术的基础知识和基本原理,包括计算机网络基础知识、IP 地址、域名和统一资源定位器 URL,Web 的基本概念、工作原理和 Web 站点的体系结构,Web 数据库基础知识等,为 Web 应用开发做好准备。

1.1 Web 简介

1.1.1 什么是 Web

Web 全称 World Wide Web,简称 WWW,译名万维网或全球信息网。

Web 提供一个图形化的界面,用以浏览网上资源。它是一个在 Internet 上运行的全球性、分布式信息发布系统。该系统通过 Internet 向用户提供基于超媒体的数据信息服务。它把各种类型的信息(文本、图像、声音和影视)有机地集成起来,供用户使用。

Web 是 Internet 提供的一种服务,是基于 Internet、采用 Internet 协议的一种体系结构。Web 技术是 Internet 的核心技术之一,它的主要功能是信息发布和信息处理,这也是网上信息系统的一项重要功能。如网上购书系统,读者在浏览器表单中输入需要购买的图书信息,Web 服务器端应用程序接受用户信息,在数据库中查询所需图书,通过与用户的交互,完成购书操作。

Web 技术几乎已进入所有信息领域,如新闻、广告、信息服务、电子商务、电子政务和企事业管理信息系统等。

1.1.2 Web 三要素

在 Web 环球信息网中遨游必备的三要素如下所示。

(1) 统一资源定位(URL):解决网上资源在何处的问题。

(2) 超文本传输协议(HTTP):解决用什么方法访问资源的问题。

(3) 超文本标记语言(HTML):提供信息资源的表达方式和在资源之间自由访问的手段。

1.2 计算机网络基础

计算机网络是计算机技术和通信技术相结合的产物。基于 Web 的信息系统是运行在计算机网络之上的。Internet 是覆盖全球、开放的、由众多网络互联而形成的计算机网络。

1.2.1 网络的定义

计算机网络：用通信线路和通信设备，将分布在不同地点的具有独立功能的多个计算机系统连接起来，在网络软件的支持下，实现彼此之间数据通信和资源共享的系统。

计算机网络常见的分类依据是网络覆盖的地理范围，根据网络覆盖范围的大小将网络分为局域网、城域网和广域网。

局域网(Local Area Network，LAN)：是连接近距离的网络，覆盖范围从几米到数千米，如办公室或实验室的网，同一建筑物内的网，校园内、某单位园区内或某居民区内的网。局域网是城域网和广域网的基础。

城域网(Metropolitan Area Network，MAN)：是介于局域网和广域网之间的一种高速网络，覆盖范围为几十千米，其规模限于一个城市的范围。

广域网(Wide Area Network，WAN)：其覆盖范围从几十千米到几千千米，可以连接若干个城市、地区、国家，甚至横跨几个洲覆盖全球，形成国际性的远程网络。广域网的连接一般采用租用线路、VPN 虚拟专用网、DDN、X.25、卫星信道和帧中继等通信线路。

1.2.2 Internet

1. Internet 定义

Internet 译为“因特网”，也称国际互联网。Internet 是一个把世界范围内的众多计算机、人、数据库、软件和文件连接在一起的，通过一个共同的通信协议(TCP/IP 协议)相互会话的网络。

该网集合了全球大量信息资源，是信息时代人们交流信息不可缺少的手段和途径。与 Internet 相连的任何一台计算机，都被称为主机。Internet 技术主要有以下几方面表现：

- 采用标准协议——TCP/IP 协议，可使网上各种不同的计算机进行通信。
- 通过路由器将不同网络互联。
- 提供了建立在 TCP/IP 协议基础之上的 WWW 浏览服务。
- 应用 DNS 域名解析系统完成网络计算机之间的地址解析工作。

Internet 可以定义为使用 TCP/IP 协议由路由器连接起来的覆盖全球的网络系统。

2. Internet 提供的基本服务

Internet 已经深入到人们生活、工作、学习、娱乐的各个方面,已开发了大量的 Internet 应用。它提供的主要服务如下。

1) WWW 服务

WWW 是 Internet 上最方便、最受欢迎的信息服务类型。它是在 Internet 上运行的信息服务系统,WWW 上集中了全球的信息资源,信息分布在世界各地的联网主机上。WWW 提供信息服务、在线影片、音乐欣赏、网络传真和网上购物等服务。

2) E-mail(Electronic Mail)服务

电子邮件又称电子信箱,简称 E-mail,是 Internet 提供的一项基本服务,也是 Internet 上使用最广泛的一种服务。它可以发送文本文件、图片和程序等。它是网上的邮政系统,是一种以计算机网络为载体的信息传输方式。通过电子信箱地址,人们可以在互联网上快速、简便、廉价地交换电子邮件。

3) 文件传输(File Transfer Protocol,FTP)服务

文件传输服务也是 Internet 的基本功能之一,FTP 服务允许 Internet 用户将一台计算机上的文件传送到另一台计算机上。相当于每台联网的主机都拥有了一个巨大容量的备份文件库。FTP 可以在 Internet 上传输任何类型的文件,例如:文本文件、二进制文件、图像文件、声音文件和数据压缩文件等。FTP 服务有两种类型,普通 FTP 服务和匿名(anonymous)FTP 服务。普通 FTP 服务向注册用户提供文件传输服务;匿名 FTP 向任何 Internet 用户提供文件传输服务。

4) 远程登录(Telnet)服务

在网上人们常常需要使用远程计算机的资源,为此人们开发了远程终端协议,即 Telnet 协议。Internet 用户可以使用 Telnet 命令,使自己的计算机进入远程主机系统。使用 Telnet 命令与远程主机建立连接后,用户就像坐在远程主机面前一样,使用远程主机的资源和应用程序。

5) BBS(Bulletin Board System)服务

BBS 是 Internet 上的一种电子信息服务系统。它提供一块公共电子白板,每个用户都可以在上面发布信息并提出自己的观点。电子公告栏可以按不同的主题、分主题形成多个布告栏。BBS 允许用户上传和下载文件,讨论和发布通告等,是网民们交流信息、讨论问题的理想场所。

6) 新闻(Usenet)服务

Usenet 是针对某个主题的网上新闻组。新闻组可以使趣味相投的人们通过电子邮件和电子公告栏讨论共同关心的问题。加入某个新闻组后,可以浏览新闻组中的文章,回复他人的文章,也可以发布自己的文章。

7) 网上电话与网上视频服务

在 Internet 上打电话,费用比一般的电话要少得多。网上电话可使用普通电话机、专用的 IP 电话机,也可以使用多媒体计算机代替电话机。如果使用多媒体计算机来打电

话，在使用前，需要正确安装声卡、音箱、麦克风并运行相应软件（例如 IP Phone）。

利用 Internet，还可以收发传真，在高速宽带的网络环境下收看广播视频节目、举办远程视频会议、远程教学、远程诊断等。

8）电子商务（E-Commerce）

在 Internet 上可以实施基于 Web 的商务活动，任何通过 Internet 进行的产品或服务的销售行为，都属于电子商务范畴，这些商务活动可以有 B2B（商家对商家）、B2C（商家对客户）、B2G（商家对政府）等多种形式。网上在线交易方便，价格低廉，是 Internet 上增长很快的领域。

9）电子政务（E-Government）

电子政务是一个广义的词汇，主要用于网上办公、网上审批、信息发布和应急指挥等。电子政务的核心价值在于改进办事方式，提高工作效率和推动职能转变等。

随着 Internet 的发展，它提供的服务还在不断增长，应用领域也在不断扩大。

3. Internet 体系结构

因特网是由分布在世界各地的网络系统通过通信介质（光纤、电缆、微波和卫星等）及网络设备连接起来的彼此可以交流信息、规模巨大的网络系统。图 1-1 给出了 Internet 示意图。

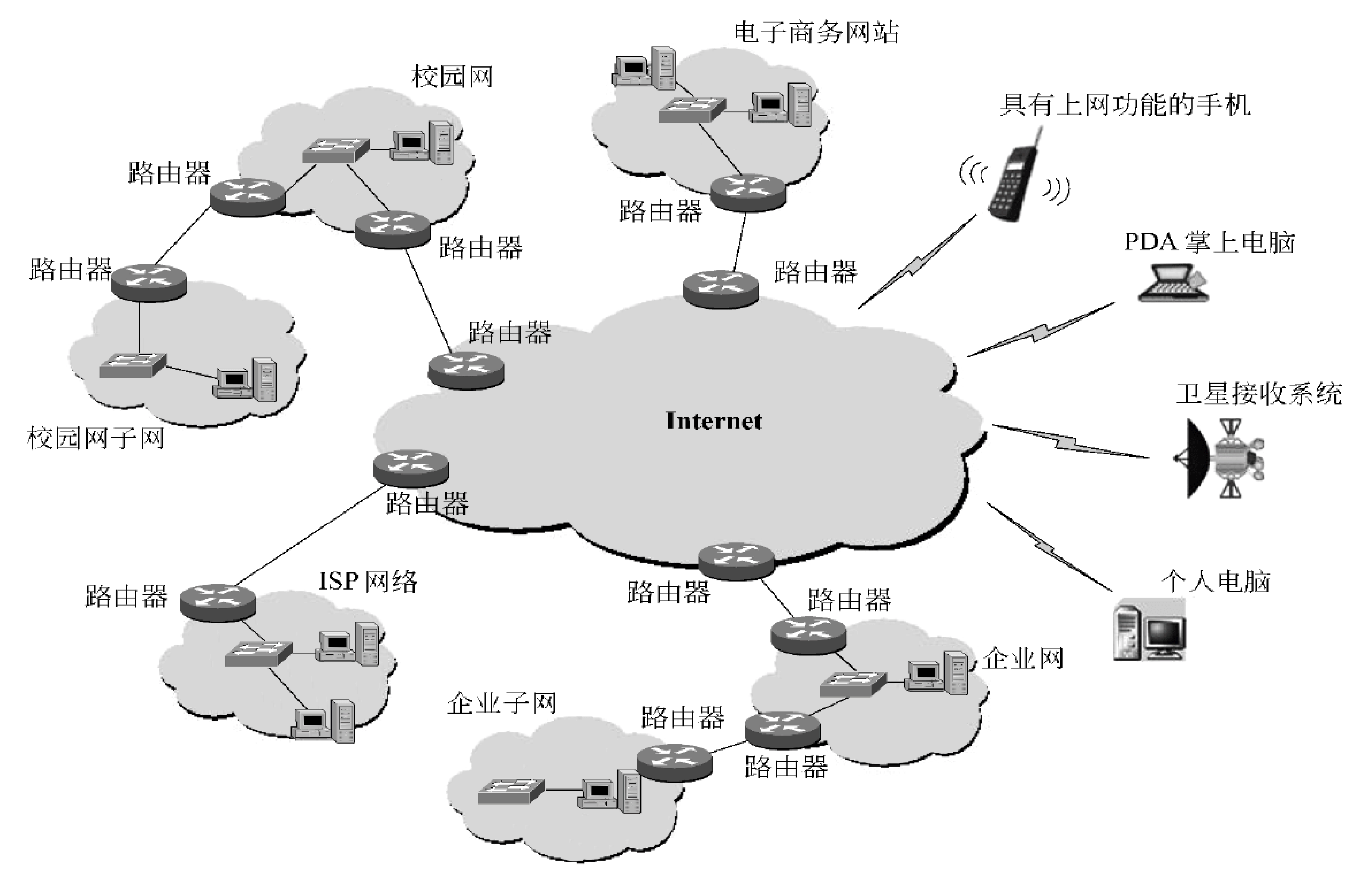


图 1-1　Internet 示意图

在图 1-1 中，路由器是 Internet 实现互联的"标准件"，路由器能够将使用不同技术的两个网络互联起来，它的主要作用是将网络互联并且执行路由选择。可以说 Internet 是一个由路由器连接起来的网联网。

1.2.3 OSI 参考模型

为共享计算机网络的资源，在网上交换信息，需要实现不同系统中的实体间的通信。实体包括应用程序、文件、数据库、电子邮件、服务器及终端等。计算机之间的数据通信必须遵守某种约定和规程，这些约定和规程就是网络通信协议。协议的三个要素如下。

① 语法(syntax)：数据和控制信息的结构或格式，即对信息的数据结构做一种规定。例如，用户数据和控制信息的结构和格式等。

② 语义(semantics)：对协议元素的含义进行解释，不同类型的协议元素规定的语义是不同的。例如，需要发出何种控制信息，完成何种动作和是否得到响应等。

③ 时序(timing)：实体通信实现顺序的详细说明。例如，双方进行通信时，发送点发出一个通信报文，如果目标点正确收到，则回答发送点接收正确；若收到错误信息，则要求发送点重发一次。

国际标准化组织(ISO)提出的"开放系统互连模型"是计算机网络通信的框架模型。TCP/IP 是 Internet 使用的通信协议。

OSI(Open Systems Interconnection)指开放系统互连，是 ISO(国际标准化组织)制定的网络系统框架结构，是一切网络互联的基础模型。OSI 参考模型采用了分层的结构化技术。层次的划分从逻辑上将功能分组。层次要足够多，以使每一层小到易于管理；但也不能太多，使汇集各层的处理开销太大。OSI 采用了 7 层体系结构，各层完成一组特定的任务，每层直接为其上层提供服务。OSI 参考模型层次划分遵循的原则如下：

- 网络中各结点具有相同的层次，相同的层次具有相同的功能。
- 同一结点内相邻层之间通过接口通信。
- 每一层使用下层提供的服务，并向其上层提供服务。
- 不同结点的同等层按照协议实现对等层之间的通信。

OSI 参考模型各层功能简介见表 1-1。

表 1-1 OSI 参考模型

层号	层的名称	层的英文名称	功 能 简 述	
7	应用层	Application Layer	网络应用进程层：在用户进程之间交互用户信息，直接为用户提供服务	上层
6	表示层	Presentation Layer	数据表示：处理两个通信系统中交换信息的表示方式，对用户数据进行格式转换、数据加密与解密、数据压缩与恢复等	
5	会话层	Session Layer	主机间通信：提供两进程间建立、维持和终止会话连接功能；支持交互会话管理功能	
4	传输层	Transport Layer	端对端的连接：提供建立、维护和拆除传送连接的功能；提供端到端的错误恢复和流量控制	

续表

层号	层的名称	层的英文名称	功 能 简 述	
3	网络层	Network Layer	选择最优路由：控制分组传送系统的操作、路由选择、拥挤控制和网络互联	下层
2	数据链路层	Data Link Layer	接入介质：提供网络层实体间数据发送和接收的功能和过程；提供数据链路的流量控制	
1	物理层	Physical Layer	二进制传输：指定传输方式的要求；提供为建立、维护和拆除物理链路所需要的机械的、电气的、功能的和规程的特性；故障检测指示	

在 OSI 参考模型中，网络层以下称为下层协议（也称通信协议），用于创造两个网络设备间的通信连接；传输层以上的 4 层称为上层协议（也称应用协议），主要负责两个网络设备间的互操作。

1.2.4 TCP/IP 协议

TCP/IP 是英文 Transportation Control Protocol/Internet Protocol 的缩写，意思是传输控制协议/网际协议。TCP/IP 协议是实现国际互联网的连接性和互操作性的关键协议，它就像胶水一样把 Internet 上成千上万的网络互联起来，是 Internet 上所有计算机进行信息交换和传输所采用的协议，也是 Web 服务器与其他网络计算机互联的基本通信协议。Internet 的应用层为用户提供了各种服务，例如要访问 Web 网站，可输入域名，然后由 DNS 服务将域名解析成 IP 地址；应用 FTP 传输文件；应用 Telnet 协议登录到远程计算机中；应用 SMTP 协议发送电子邮件等。

TCP/IP 协议拥有一套完整而系统的协议标准，它实际上是由一组协议构成，称为 TCP/IP 协议族，其中最主要的两个协议是 TCP 协议和 IP 协议。TCP/IP 协议覆盖了 OSI 协议的全部内容，因此也有人称其为 TCP/IP 体系结构，如图 1-2 所示。TCP/IP 协议对标准的 OSI 七层协议进行了简化，它没有表示层和会话层，这两层的功能由应用层提供。TCP/IP 协议一般分为 4 层，其功能见表 1-2。

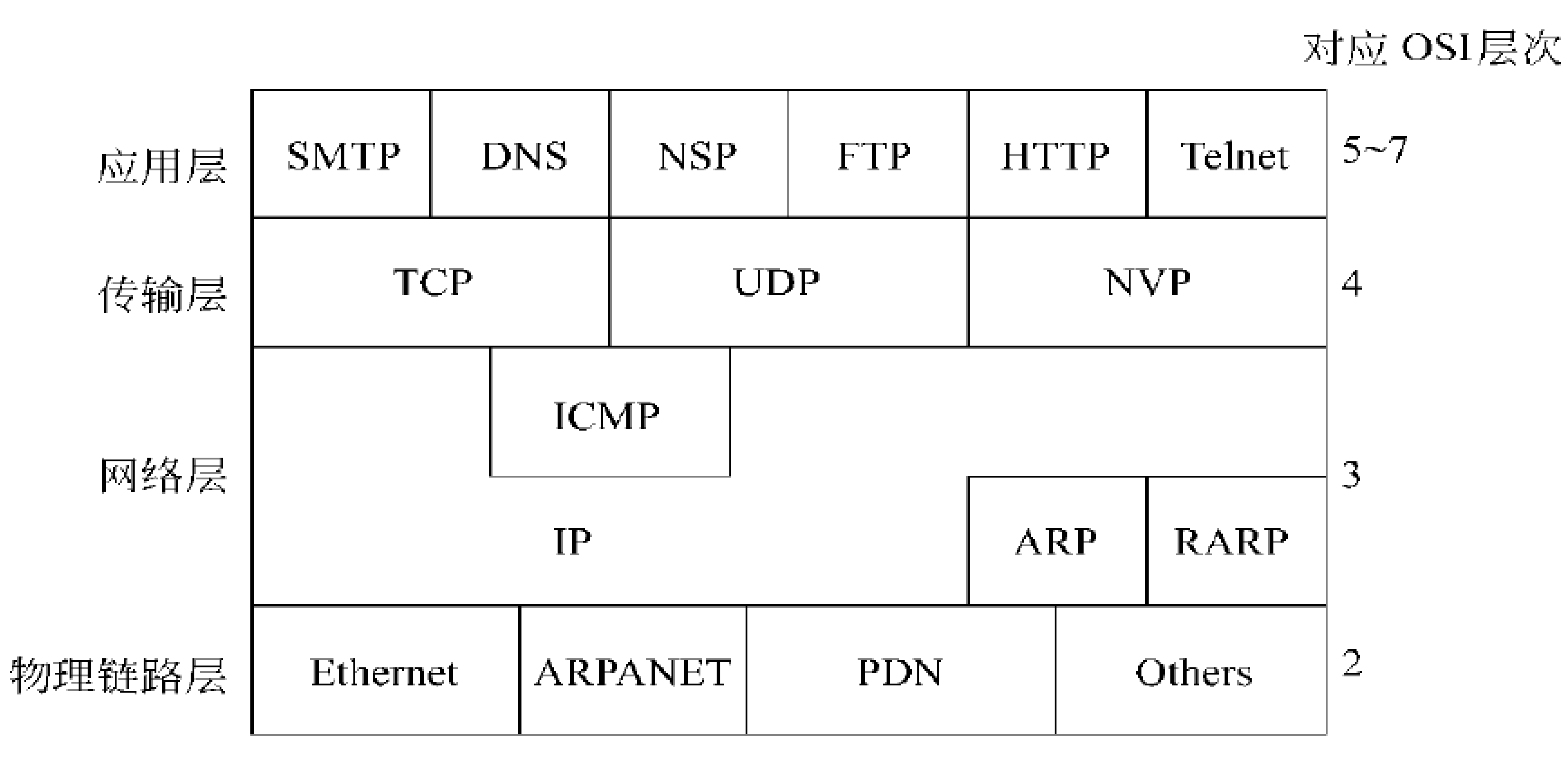


图 1-2　互联网协议层次结构

表 1-2　TCP/IP 各层的主要功能

层 的 名 称	功 能 简 述
应用层	向用户提供一组常用的应用程序，如文件传输、电子邮件等
传输层（TCP）	提供端到端的数据传输服务
网络层（IP）	定义数据报，处理路由
物理链路层	接收网络层数据报，通过网络发送；从网络上接收数据送交网络层

TCP/IP 协议的主要功能如下。

1. 应用层

应用层是 TCP/IP 协议的最高层，它提供一些常用应用程序，主要服务如下。

Telnet：虚拟终端协议，提供远程登录功能。

FTP：文件传输协议，用来控制两台主机之间文件的交换。

SMTP：简单电子邮件传输协议，实现互联网中电子邮件收发功能。

DNS：域名服务协议，用于实现网络设备域名到 IP 地址的转换功能。

HTTP：超文本传输协议，在客户机浏览器和服务器之间传输 Web 文档。

2. 传输层

传输层也称 TCP 层，主要负责应用进程之间端到端的通信。

TCP：传输控制协议。TCP 协议把数据分成若干个数据包，给每个数据包写上序号，以便接收端将数据恢复成原来的格式。

UDP：用户数据报协议，它是一个面向无连接的协议。UDP 协议不采用复杂的数据可靠性保护机制，不支持数据丢失和数据报重传处理。如果需要可靠数据传输时，不能使用 UDP 协议，而应采用 TCP 协议。

NVP：网络语音协议。

3. 网络层

网络层也称 IP 层，负责互联网中计算机之间的通信。

IP：网际协议，IP 协议给每个数据包写上发送主机和接收主机地址，负责在网络上传输由 TCP 或 UDP 装配的数据包。它的主要功能是：管理 Internet 中的地址；路由选择，在源方和目的方之间选择一条最佳路径；数据报的分片与重组。

ICMP：控制报文协议，用于报告差错和传输控制信息。

ARP：地址转换协议，负责将 IP 地址转换为计算机物理地址。

RARP：反向地址转换协议，将计算机的物理地址转换为 IP 地址。

4. 物理链路层

物理链路层的主要功能是：接收网络层的 IP 数据报，通过网络向外发送；接受处理

从网络上传来的物理帧，抽出 IP 数据包，向网络层发送。它是主机与网络的实际连接层。

1.3 IP 地址、域名和 URL

Internet 对网络资源的定位是由一长串数字（IP 地址）来实现的，由于 IP 地址不易记忆，人们使用域名解析系统，为每台主机指定易于记忆的名字（主机名/域名）与 IP 地址对应。也就是说，网上的主机既可以使用 IP 地址定位也可以使用主机名/域名定位。

1.3.1 IP 地址

1.3.1.1 IP 地址的作用

Internet 地址又称 IP 地址，通过它可以唯一确定 Internet 上每台主机或设备。Internet 上每台主机或设备都有一个唯一的地址以确定它在何处。

在 TCP/IP 协议中分配给每台主机一个 32 位二进制数作为该主机的 IP 地址，在 Internet 上发送的每个数据包都包含了一个 32 位的发送方地址和一个 32 位的接收方地址。

1.3.1.2 IP 地址表示法

IPv4 地址采用“点分十进制”表示法。它由一个 4 个字节 32 位的二进制数组成，对应 4 组十进制数。例如：

二进制表示	11001010	01100000	00111101	10101000
4 组十进制数 IP 地址	202 .	96 .	61 .	168

IP 地址由网络地址和主机地址两部分组成。网络地址标识该主机所在的网络，主机地址标识该主机在该网络中的位置。

网络地址（netid）	主机地址（hostid）

IP 地址的层次结构具有两个重要特性：

- 为每台主机分配了一个唯一的地址。
- 网络号必须全球统一分配，主机号由本地分配，不需要全球统一分配。

为解决 IPv4 的 32 位地址不够使用问题和分配不均问题，人们又开发了 IPv6。IPv6 地址是 128 位二进制数字，采用冒号分割的十六进制数表示，大大扩充了 IPv4 的地址空间，目前在试运行阶段。

1.3.1.3 5 类 IP 地址

网络的 IP 地址有 5 类（A、B、C、D、E）。其中 A、B、C 是基本网络地址格式，A 类地址格式用于大型规模网络，B 类地址用于中型规模网络，C 类地址用于较小规模网络，D

类和 E 类地址是一些特殊类型的 IP 地址。IP 地址格式见表 1-3。

表 1-3　IP 地址格式

字节 / 位 / 类别	字节 1								字节 2	字节 3	字节 4
	0	1	2	3	4	5	6	7	8　15	16　23	24　31
A 类	0	网络地址(7 位)							主机地址(24 位)		
B 类	1	0	网络地址(14 位)							主机地址(16 位)	
C 类	1	1	0	网络地址(21 位)							主机地址(8 位)
D 类	1	1	1	0	组播地址(28 位)						
E 类	1	1	1	1	保留地址(供实验和将来使用)						

IP 地址范围见表 1-4。

表 1-4　IP 地址范围

类别	地 址 范 围	最大网络数目	最多主机数目
A	1.0.0.0～126.255.255.255	126	16 777 214
B	128.0.0.0～191.255.255.255	16 384	65 534
C	192.0.0.0～223.255.255.255	2 097 151	254

1.3.1.4　几个特殊意义的 IP 地址

以下几个有特殊意义的 IP 地址不能用于网络的主机地址。

1. 广播地址

主机地址位全是 1,表示网上所有的主机,可以向网上所有的主机发送信息。例如,148.08.255.255 表示向 148.08 网上的所有主机发信息。

2. 本地网络地址

IP 地址中主机地址位都是 0,表示本地网络地址。例如 148.08.0.0 表示一个 B 类网络地址 148.08。

3. 回放地址

127.0.0.1 称为回放地址,用于网络软件测试及本地机进程间通信的地址。应用程序发往该地址的信息被交回给应用程序,不进行任何网络传送。

4. 内部保留地址

保留给内部网络使用的地址,企业建立内部网络时可以使用这些地址。由这些地址

发往 Internet 的信息必须经过地址转换成标准的 Internet 地址后才能在 Internet 上传送这些地址，它们包括：10. *. *. *，192.168. *. *，172.16. *. *（* 为 0～255 之间任一数字）。

主机标识的各个位不能都设置为 1，也不能都设置为 0。A 类地址 0. *. *. * 对应默认路由器。

1.3.1.5 子网掩码

把 A、B 或 C 类网进行子网划分，可以充分利用 IP 地址资源，为更多的主机分配 IP 地址。子网划分把网络地址的两级结构转换成三级结构，如图 1-3 所示。

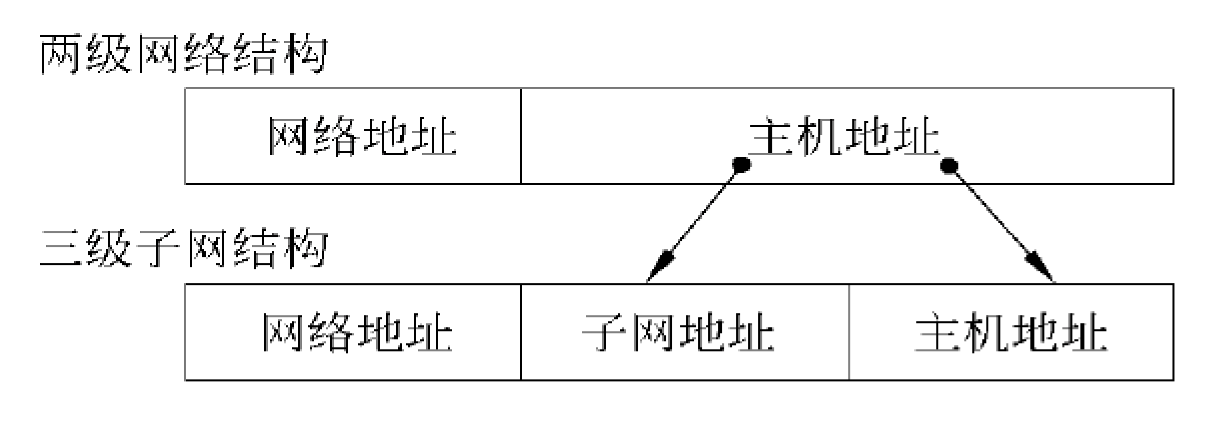


图 1-3 三级子网结构图

子网掩码可以将一个 IP 地址分解为对应的网络地址、子网地址及主机地址。子网掩码是一个与 IP 地址等长的二进制数，经过与 IP 进行“与”操作后得出网络地址、子网地址和主机地址。例如，已知 IP 地址和子网掩码及其对应的十进制数如下：

IP 地址：202.204.224.198 11001010 11001100 11100000 11000110

子网掩码：255.255.255.192 11111111 11111111 11111111 11000000

运算结果的网络地址是 202.204.224，子网地址为 3，主机地址是 6。

1.3.1.6 IP 地址的使用与企业网 IP 地址规划

1. 根据 IP 地址判断其网络类别、网络地址和主机地址

已知主机的 IP 地址为 206.196.0.133，子网掩码是 255.255.255.0，请确定该主机所在网络的类别、网络号及它的主机号。判断步骤如下。

1）把 4 组十进制数转变为 4 字节 32 位的二进制数

4 组十进制数： 206 . 196 . 0 . 133

32 位二进制数： 11001110 11000100 00000000 10000101

2）确定网络类别

对照表 1-3，第 1 字节是 11001110，它的第 0、1、2 位是 110，所以该主机所在网络的类别是 C 类。

3）确定网络地址

C 类网的前 3 个字节是它的网络地址：206.196.0。

4）确定主机地址

C 类网的主机地址是第 4 字节，所以它的主机地址是 133。

结论：该主机是 C 类网 206.196.0 中的 133 号主机。

2. 根据 IP 地址和子网掩码判断其网络类别、网络地址、子网地址和主机地址

假设已知网络 IP 地址和子网掩码如下。

IP 地址　　　　11000000.01001111.00101110.01100001＝198.79.46.97

子网掩码　　11111111.11111111.11111111.11100000＝255.255.255.224

试确定该主机所在网络的类别、网络号、子网号及它的主机号。判断步骤如下。

(1) IP 地址的前三位是 110,说明该地址是一个 C 类地址。

(2) 前三个字节标识网络地址,网络地址是 198.79.46。

(3) 后一个字节标识主机,对照子网掩码的最后一个字节,前三位是 1 后五位是 0,所以子网地址编号占三位,主机地址占 5 位。根据 IP 地址最后一字节的前 3 位是 011,后 5 位是 00001,所以辨认出子网编号是 3,主机编号是 1。

结论:IP 地址 198.79.46.97 标识的是 C 类网络 198.79.46 的 3 号子网的 1 号主机。

3. 为单位规划 IP 地址

请为管理学院规划 IP 地址,该学院有 6 个局域网,每个局域网最多有 26 台主机(或网络设备)。规划过程如下所示。

1) 申请 IP 地址

管理学院最多有 6 个局域网 156 台主机,若为 6 个局域网申请 6 个 C 类 IP 地址,共有 6×254＝1524 个 IP 地址,实际使用 156 个地址,将有 1368 个 IP 地址的浪费。实际应用中,可以使用子网的方法,使这 6 个局域网共用一个 C 类网的地址。把这 6 个子网当做一个整体,申请一个 C 类 IP 地址。假设管理学院申请到的 C 类 IP 地址是 202.224.46。

2) 确定子网地址的位数与子网地址

子网地址用于标识管理学院内部的不同子网。C 类网的主机地址占 8 位,由于该学院有 6 个局域网,子网地址应占 3 位形成 8 个网段(000～111),其余 5 位是子网中的主机地址,每个子网可以有 30 个主机地址,对于该学院也够用了。图 1-4 说明了管理学院子网位数与主机地址位数的分配。

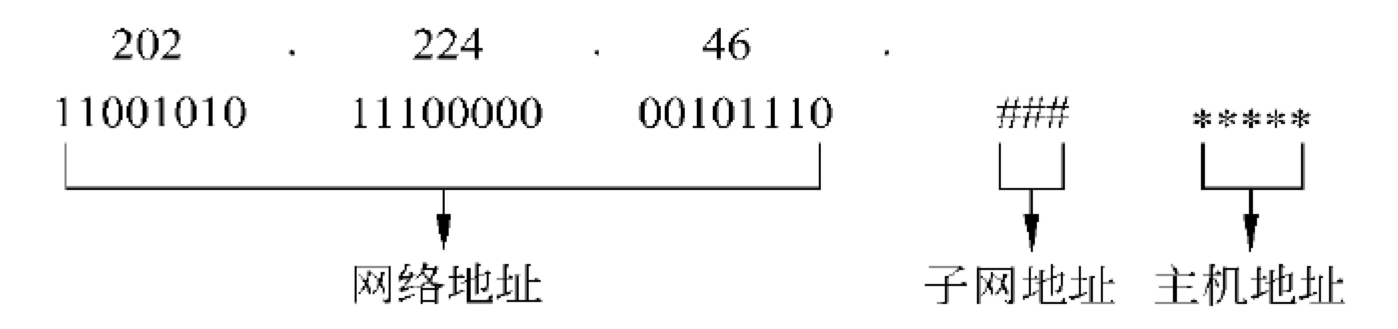


图 1-4　子网地址位数与主机地址位数的分配

以下列出各子网地址:

1 号子网地址　　11001010 11100000 00101110 00000000＝202.224.46.0

2 号子网地址　　11001010 11100000 00101110 00100000＝202.224.46.32

3 号子网地址　　11001010 11100000 00101110 01000000＝202.224.46.64

4 号子网地址　　11001010 11100000 00101110 01100000＝202.224.46.96

5 号子网地址　11001010 11100000 00101110 10000000＝202.224.46.128

6 号子网地址　11001010 11100000 00101110 10100000＝202.224.46.160

3）主机地址分配方案

以 1 号子网为例，说明主机地址的分配。

1 号主机地址　11001010 11100000 00101110 00000001＝202.224.46.1

2 号主机地址　11001010 11100000 00101110 00000010＝202.224.46.2

3 号主机地址　11001010 11100000 00101110 00000011＝202.224.46.3

⋮　⋮　⋮　⋮　⋮　⋮

26 号主机地址　11001010 11100000 00101110 00011110＝202.224.46.26

4）子网掩码的确定

管理学院的子网掩码是：

11111111.11111111.11111111.11100000＝255.255.255.224

1.3.2　域名

网上主机的 IP 地址是一个 4 组十进制数，使用起来很不方便，人们又为网上的主机指定了一个易于记忆的域名，并开发了一个域名解析系统（Domain Name System，DNS），使每台主机的主机名/域名与它的 IP 地址对应。当用户与 Internet 上某台主机交换信息时，只需要使用主机名/域名，网络会自动将其解析成 IP 地址，找到这台主机。

域名的结构：主机名.三级域名.二级域名.顶级域名

域名分为多个区域，从左到右表示的区域范围越来越大。“一级域名”又称顶级域名，代表国家和组织，如我国的顶级域名是“cn”。“cn”下的二级域名分为类别域名和行政区域名两类。类别域名有 6 个，它们是 ac（科研机构）、com（商业组织）、edu（教育机构）、gov（政府部门）、net（互联网络、接入网络的信息中心和运行中心）、org（各种非盈利性组织），类别域名还在增加，如新增加的 TV 域名等。行政区域名有 34 个，也就是我国的 34 个省市。三级域名通常是组织机构名，一般以各单位或机构名字的英文简写命名。CERNET 网络中心负责二级域名“edu”下三级域名的注册申请，中国互联网信息中心（CNNIC）负责其余 39 个二级域名下的三级域名申请。“主机名”是第四级域名，用有意义的英文名称代表网络上的主机。根据主机所在域的不同，域的级别可能多于或少于四级。

例如，域名 www.people.com.cn，其中，cn 为顶级域名，表示中国；com 是二级域名，表示商业组织；people 是三级域名，组织结构名，表示人民网；www 是主机名，表示人民网的 www 主机。又如，域名 www.e-gov.org.cn 表示中国电子政务网。

地址解析从下至上逐级进行，当某一联网单位内的主机访问互联网上的资源时，先由本单位的 DNS 解析其地址，如果该地址在本地 DNS 中找不到，则将此地址交给上级的 DNS，逐级上推，直到互联网的根 DNS，如果还不能找到，说明所要求的是一个不存在的地址。

1.3.3 URL

URL(Uniform Resource Locator)称为统一资源定位器,用来指明网络资源在Internet上的位置,它的功能相当于通信地址。URL能以唯一且一致的方式定义每个资源在Internet上的位置。

1. URL格式

URL的格式为:

<协议>://<主机名:端口号><文件路径>
(访问协议)　(资源位置)

一个URL的例子如:http://www.bta.net.cn:80/software/home.html。

1) 协议

协议表示取得资源的方法或通信协议的种类。例如,http。最常用的是HTTP协议,其他常见的有news、file、Telnet、FTP、Gopher等。

2) ://

://是URL规范要求的标记。

3) 主机名

主机名是要访问的服务器的全名(服务器全名包括域名和主机名),也可以是服务器的IP地址,表明服务器在网络中的位置。例如,www.bta.net.cn。

4) 端口号

对某些资源访问,需要给出相应服务器提供的端口号,以使操作系统用来辨别特定信息服务的软件端口。例如,HTTP的标准端口号是80。一般情况下服务器程序采用标准的保留端口号,所以可以省略端口号。

5) 文件路径

文件路径是服务器上保存目标文件的目录,它指定浏览器访问的最终目标。例如,software/home.html。

2. URL的使用

例如,Internet上某一资源的URL是http://www.bta.net.cn/software/home.html,该URL告诉浏览器要访问的文件类型是HTML文件,使用HTTP协议,在名为www.bta.net.cn的服务器上,访问/software/目录下名为home.html的文件。如下是某些URL的例子:

```
telnet://odysseu.bbb.com:70
http://www.buu.com.cn:8080
ftp://ftp.w3.org/pub/www/doc
```

3. 文件定位的几种方式

文件定位可以有3种方式:域名方式、IP地址方式和文件目录方式。可以在浏览器

的地址栏目里使用这3种方式查询信息。例如，某服务器的域名是www.bta.net.cn，IP地址是202.106.196.56。在地址栏目输入www.bta.net.cn和202.106.196.56都可以看到该服务器的默认主页。如果用浏览器查看本机的文件，在地址中输入文件名全名，如C:/webshare/wwwroot/homepage.html，就可以看到homepage主页。

1.4 Web基础知识

1.4.1 Web工作机制

Web的结构及其工作机制见图1-5，它的工作过程如下：

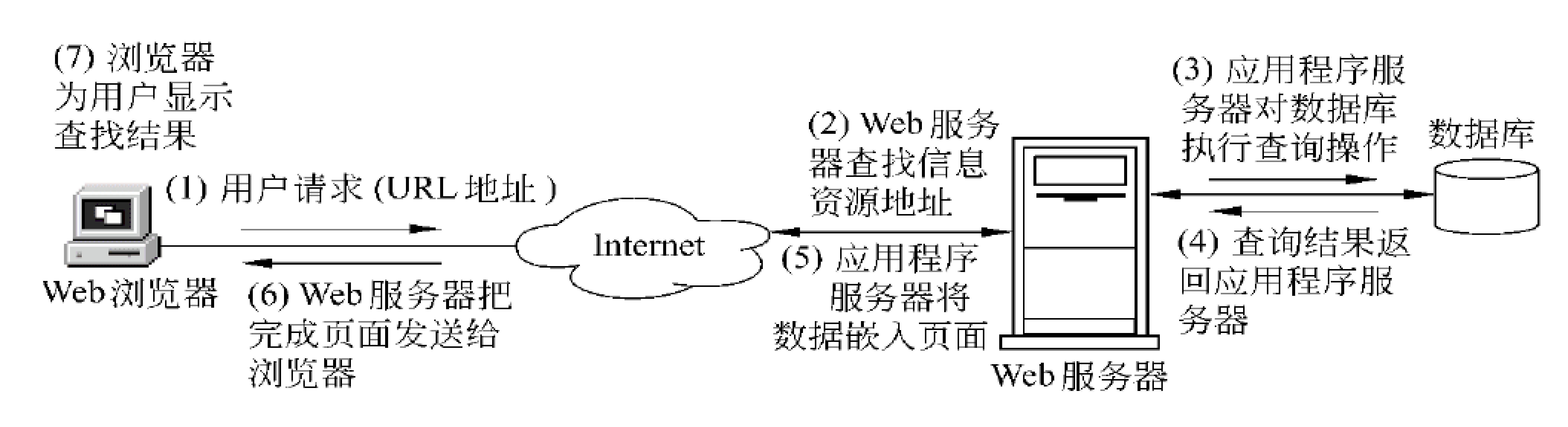


图1-5 Web的工作过程

(1) 启动客户端浏览器，在浏览器中确定将要访问页面的URL地址。经地址解析，找到服务器IP地址，向该地址所指向的Web服务器发出请求。

(2) Web服务器根据浏览器送来的请求，把URL地址转换成页面所在服务器上的文件全名，找到相应的文件。

(3) 如果URL指向HTML文档，Web服务器使用HTTP协议把该文档直接送给浏览器。如果HTML文档中嵌入了CGI、ASP/ASP.NET、PHP或JSP程序，则由Web服务器运行这些程序，把结果送到浏览器。如果应用程序包含有对数据库的操作，则应用程序服务器将查询指令发送给数据库驱动程序，由数据库驱动程序对数据库执行操作。

(4) 查询结果返回给数据库驱动程序，并由驱动程序返回Web服务器。

(5) Web服务器将结果数据嵌入页面。

(6) Web服务器将完成的页面发送给浏览器。

(7) 浏览器解释HTML文档，在客户端屏幕上显示结果。

1.4.2 Web站点体系结构

1.4.2.1 浏览器/应用服务器/数据库服务器三层结构

网络信息服务在逻辑上采用浏览器/服务器(Browser/Server，简称B/S模型)工作模式，一般用户的计算机称为客户机，用于提供服务的机器称为服务器。

浏览器/服务器的体系结构可划分为二层结构和三层结构，基于 Web 的数据库应用采用三层浏览器/服务器结构，也称 Browser/Server/Database Server 结构(见图 1-5)。第一层浏览器，第二层 Web 服务器，第三层数据库服务器。浏览器是输入数据和显示结果的用户界面，在浏览器表单中填入数据，单击提交按钮，表单中的数据被发送到 Web 服务器。Web 服务器端应用程序接受并处理用户数据，并从数据库中查询用户数据或把用户数据录入数据库。最后，Web 服务器把返回结果插入 HTML 页面，传送到客户端，在浏览器中向用户显示。

三层结构的优点在于：

- 应用系统处理逻辑与数据库系统分开，数据库内容更新不影响应用系统处理逻辑。
- 专门的应用服务器处理客户请求，并与数据库通信，提高了数据库的访问效率。
- 数据处理和数据库操作任务移到服务器执行，客户机上只要能够运行浏览器即可。

1.4.2.2 Web 站点的集成

Web 站点的集成示意图见图 1-6。虚线之内代表一个企业或 ISP 的内部网络。一般来说企业网有自己的服务系统，它的企业信息管理系统、企业办公系统、数据库等服务是为企业内部服务的。防火墙将内部网与因特网隔离，保证内部网络系统的安全性。

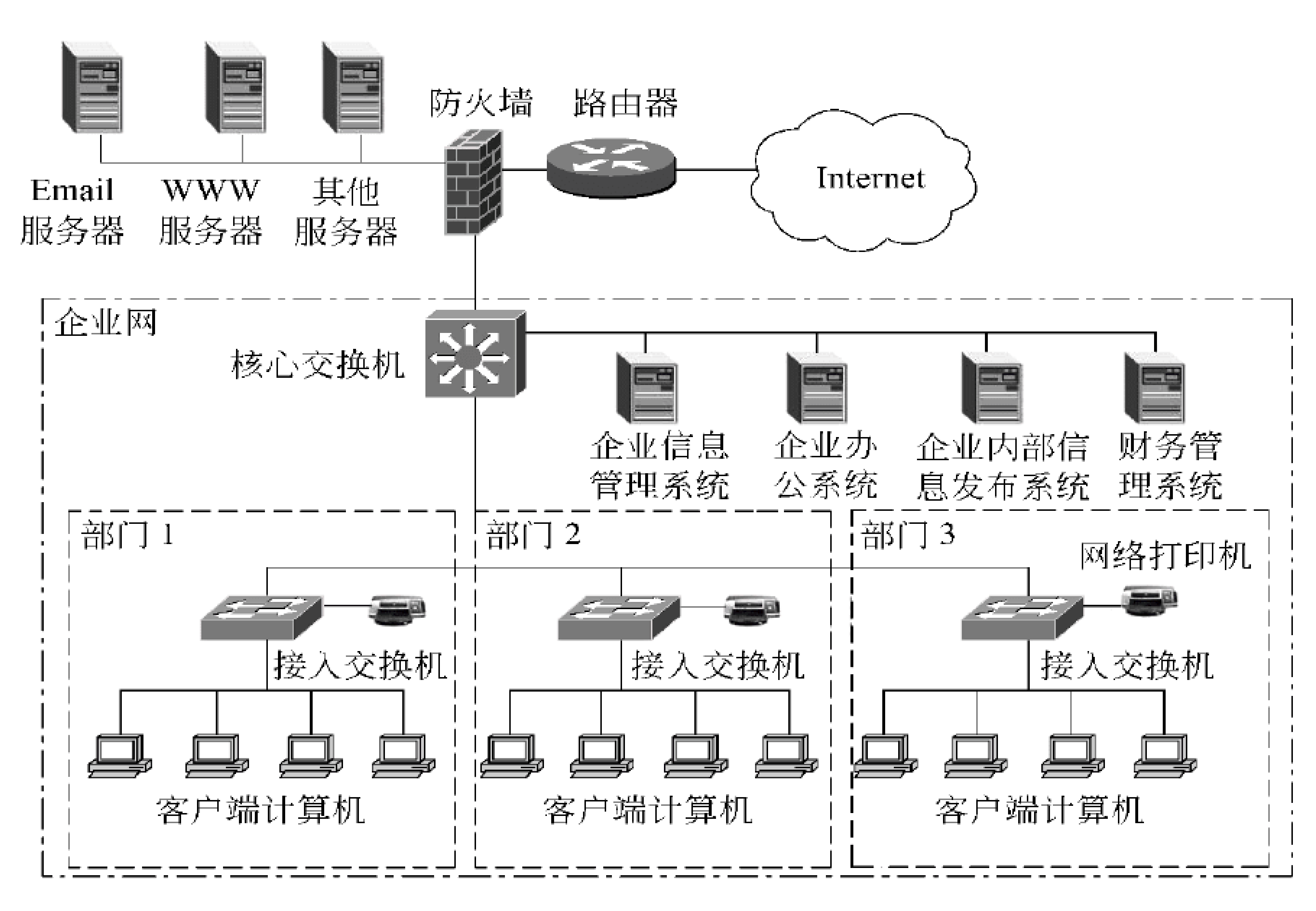


图 1-6 Web 站点集成示意图

虚线之外部分是该企业的网络系统提供的对外的 Mail、WWW、FTP、DNS 等服务功能，这些服务器或放在防火墙的外面，或放在防火墙的非军事化区。它们的地址对外部用户是可见的，保证外部用户对这些服务器的访问。整个企业网通过路由器与因特网连接。

企业服务器一般包括：WWW 服务器，用于企业信息的对外发布；Mail 服务器，用于企业的电子邮件系统与外部的连接；DNS 服务器为企业用户提供因特网域名解析服务，使用户通过域名访问因特网中的站点。FTP 服务器为因特网用户提供文件或软件的上传与下载。

1.4.3 Web 应用开发技术概述

1.4.3.1 Web 客户端开发技术

1. HTML(HyperText Markup Language，超文本标记语言)

HTML 是 Web 技术的基础，它是一种文本形式的标记符号语言，可以使用任何文字处理软件编辑处理。现已有多种工具可以完成可视化的编程任务，例如，Dreamweaver 等。

2. CSS(Cascading Style Sheet，层叠样式表)

CSS 应用对页面结构的风格进行控制的思想，控制整个页面的风格。在 HTML 基础上，使用 CSS 不但能够统一、高效地组织页面上的元素，还可以使页面具有多样的外观。

3. 脚本

脚本包括 JavaScript 和 VBScript。JavaScript 是 Sun 和 Netscape 公司开发的一种解释性脚本语言，不需要编译，可以直接插入 HTML 文档中。它比 Java 更简单有效，且具有 Java 的许多特性。JavaScript 必须嵌入到 HTML 文档中，随同页面下载到客户端，由浏览器解释执行。使用 JavaScript 很容易设计与用户交互的界面。

VBScript 是由 Microsoft 公司推出的 Web 页面编程语言，它也是一种脚本语言，并继承了很多 VB 的语言特征。它也必须嵌入到 HTML 文档中，随同页面下载到客户端，由浏览器解释执行。VBScript 可以和 ActiveX 控件集成，用于开发交互式页面，它也能够进行服务器端的编程。

4. XML 技术(eXtensible Markup Language，可扩展的源标记语言)

XML 是可以定义其他语言的语言。它是 SGML 的一个简化子集，专门为 Web 环境而设计。应用 XML 制作页面的基本思想是：将内容与内容的显示方式分别定义，以使内容组织人员将精力集中于内容本身。

1.4.3.2 Web 服务器端开发技术

Web 服务器端应用程序的主要功能是生成和提供动态内容。主要技术有 CGI、ISAPI、JSP、ASP/ASP.NET、PHP 等。CGI 和 ISAPI 是使用较早的成熟技术，要求开发人员具有较强的编码能力，适合专业技术人员使用。目前流行的 JSP、ASP/ASP.NET

和 PHP 技术较为容易掌握，得到人们的普遍接受。ASP.NET 是 Microsoft 公司提供的新一代开发环境，它提供了面向对象的编程环境。

1. CGI 技术

CGI(Common Gateway Interface，公共网关接口)是 Web 开发应用中使用最早的一种技术。CGI 是动态 Web 页面的基础，也是 Web 与其他应用交互的基础。CGI 是扩展 Web 服务器功能的一种接口，通过它可以与访问 Web 页面的用户进行交互，也可以通过数据库编程接口与数据库服务器进行通信。

2. JSP(Java Server Pages)技术

JSP 是 Sun Microsystem 公司在 1999 年 6 月推出的一种动态网页技术。在 HTML 页面中加入 Java 程序段和 JSP 标记就构成了 JSP 网页。JSP 是基于 Java 的，用于网上应用开发。JSP 的结构与 ASP 很相似，但应用 JSP 开发的 Web 应用具有跨平台的特性，可以在大多数 Web 服务器上运行。JSP 将用户界面与底层的应用分开，使开发人员在不改变应用的情况下改变页面布局。由于 JSP 的种种优点，对于建立商业级的应用具有独到的优势，可以使 Web 开发人员轻松、高效地创建与维护动态网站。

3. ASP(Active Server Pages)与 ASP.NET 技术

ASP 技术是 Microsoft 公司在 1996 年底推出的一种运行于服务器端的 Web 应用程序开发技术。应用 ASP 技术，不需要进行复杂的编程，就可以开发出动态、交互、高性能的 Web 服务应用程序。ASP 不是一种开发语言，也不是一种开发工具，它是一种技术框架。它的主要功能是：为生成动态、交互并且高效的 Web 服务器应用程序，提供一种功能强大的技术或方法。

它的主要特点是：把 HTML/DHTML、脚本和数据库访问功能结合在一起，组成在服务器端的应用程序。

ASP.NET 建立在.NET Framework 类的基础之上，并提供了由控件和基础部分组成的“Web 程序模板”，大大简化了 Web 程序和 XML Web 服务的开发。程序员直接面对的是一组 ASP.NET 控件，而这些控件由一些文本框、下拉菜单等通用的 HTML 界面构件封装而成。这些控件运行于 Web 服务器上，并简单地以 HTML 的形式将界面发送到浏览器。相比于现有的 ADO 数据访问模型，ASP.NET 使用的是 ADO.NET 数据访问模型。

4. PHP(Personal Home Page Tools)技术

PHP 由创始人 Rasmus Lerdorf 在 1994 年提出，1995 年发布第一个公开版本。PHP 是自由软件，自发布起流行非常迅速。与 ASP 技术类似，PHP 是服务器端的 Web 应用程序开发技术。它具有多平台特性，能够无缝运行在 UNIX、Linux 和 Windows 平台上。它支持多种通用 Web 服务器(如 IIS、Apache 等)，在变换平台时，不需要改变 PHP 代码。PHP 对数据库的操作具有很强的兼容性，几乎支持所有的主流和非主流数据库，如：

Oracle、Sybase、MySQL、mSQL、PostgreSQL、informix、Solid、Access 和 dBase 等。

1.5 Web 数据库

数据库技术是管理信息系统的核心技术和基础技术，也是 Web 技术的一个重要组成。数据库是存放数据的仓库，数据库管理系统是一个系统软件，它的主要作用是科学地组织和存储信息，高效地获取和维护信息。数据库系统是指在计算机系统中引入数据库后的系统，一般由数据库、数据库管理系统、应用系统、数据库管理员和用户组成。

1.5.1 什么是 Web 数据库

通过 Web 应用程序访问的数据库称为 Web 数据库。

Web 是一个海量的环球信息网，为科学高效地组织与管理信息，Web 站点中的资源大多存储在 Web 站点的数据库中。数据库技术适用于大量数据的存储与管理，Web 具有友好的用户图形界面及简便的信息发布途径。所以 Web 技术与数据库技术的结合，将使它们的优势共同提升，既可以充分利用大量已有的数据库信息资源，又可以使用浏览器方便地应用数据库的资源。

与传统的数据库访问技术相比，Web 数据库访问技术的特点如下：

- 客户端统一的界面。在客户端使用浏览器，使用者只需要掌握浏览器界面的应用技术即可，大大降低了用户的使用难度。
- 统一的开发标准。HTML 是 Web 信息的组织方式，是一种国际标准，Web 服务器与浏览器都遵循该标准。基于数据库的应用都可以通过浏览器来实现，通过 Web 来访问数据库。开发者需要掌握的主要技术标准是 HTML，这在很大程度上降低了开发难度，同时也减少了开发成本。
- 跨平台运行。由于采用了统一的标准，使用 HTML 标准开发的数据库应用，可以跨平台运行，减少了开发的工作量。

1.5.2 Web 数据库工作机制

通常 Web 数据库技术应用三层或多层的体系结构，前端采用瘦客户机技术，通过 Web 服务器和中间件访问数据库，其体系结构见图 1-7。

中间件是 Web 服务器与数据库服务器之间的桥梁，负责它们之间的通信并提供应用程序服务。中间件可以直接调用脚本或外部程序来访问数据库，并将访问结果转换成 HTML 格式，通过 Web 服务器返回给客户端浏览器。

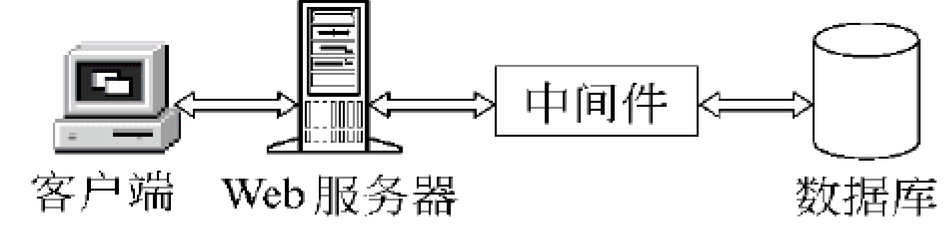


图 1-7　Web 数据库工作机制

在数据库访问应用中，一般使用 SQL (Structured Query Language，结构化查询语句)实

现对当前主流数据库的操作。

常见的数据库开发技术有：CGI 技术、服务器 API 技术、ASP/ASP. NET 技术、PHP 技术和 JSP 技术等。ASP/ASP. NET、PHP 和 JSP 也称为服务器端脚本编程技术，具有运行速度快、数据库操作功能强的特点。

1.5.3 常用 Web 数据库访问技术

1. JSP 技术

JSP 是根植于 Java 的一种动态网页制作和 Web 数据库信息发布技术。通过在 HTML 页面中嵌入 Java 程序段和 JSP 标记，实现在网页中发布数据库信息。

JSP 的主要技术特点如下：

- 具有很好的跨平台特性，一次编写，各处运行。
- 将应用程序与界面分开。
- 绝大多数 JSP 页面依赖于可重用、跨平台的组件（JavaBean 或 Enterprise JavaBeanTM 组件）执行应用程序的复杂处理。开发人员能够共享和交换组件，加速了总体开发过程。
- 数据库操作功能强大。Java 应用 JDBC（Java Database Connectivity）连接数据库，通过 JDBC 驱动程序访问数据库。Sun 公司还开发了 JDBC-ODBC Bridge，该项技术可以访问带有 ODBC 驱动程序的数据库。目前许多数据库系统带有 JDBC 和 ODBC 驱动程序。
- 继续保持 Java 的优势。

JSP 访问数据库的机制见图 1-8。

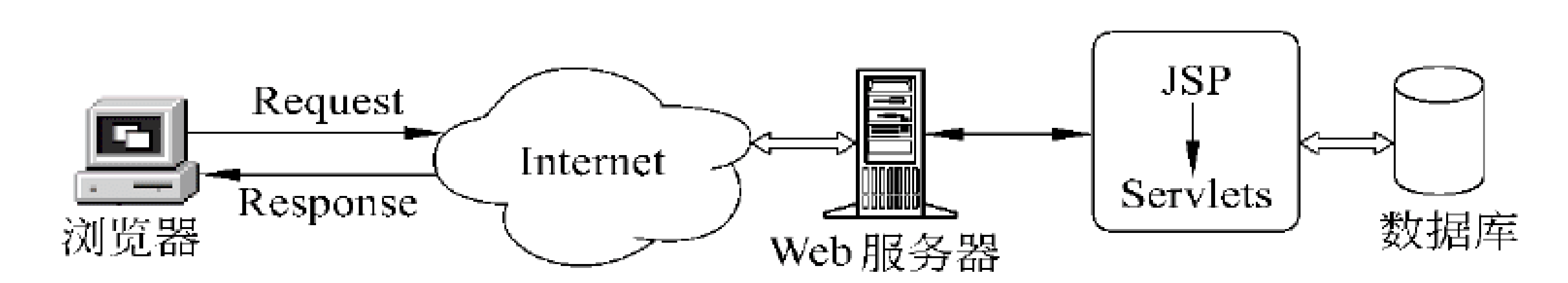


图 1-8　JSP 访问数据库的机制

JSP 的工作过程如下：

（1）在浏览器地址栏输入要请求 *.jsp 页面的 URL，发出一个 JSP 请求。

（2）Web 服务器接受扩展名为 jsp 的请求，触发 JSP 引擎。

（3）JSP 引擎检查 JSP 文件是新的还是修改过的，针对不同情况对文件进行翻译和编译，把 JSP 标记、Java 代码 HTML 内容都转换为 Servlet 代码，扩展名为 java。

（4）将产生的 Servlet 代码编译执行。

（5）将结果返回浏览器。

（6）浏览器解释执行 HTML 页面，并显示结果。

因为 Servlet 是编译过的，所以网页中的 JSP 代码不需要在每次请求该页时被解释一遍。JSP 引擎只需在 Servlet 代码最后被修改后编译一次，然后这个编译过的 Servlet 就

可以被执行了。由于是 JSP 引擎自动生成并编译 Servlet,不用程序员动手编译代码,所以 JSP 具有高效性和快速开发所需的灵活性。

2. ASP 技术

ASP 是开放式 Web 服务器应用程序开发技术,它是一种技术框架,是一种服务器端的脚本运行环境。

ASP 的主要功能是:生成动态、交互式的 Web 服务器应用程序。它能够把脚本、HTML、组件和 Web 数据库访问功能结合在一起,形成一个在服务器端运行的应用程序,并把结果转换成标准的 HTML 页面返回客户端。ASP 通过 ADO(ActiveX Data Object)访问数据库。ASP 使用脚本语言进行 ASP 程序的开发,例如 VBScript 和 JavaScript 等,在微软的 ASP 部件中,内置了 VBScript 和 JavaScript 的解释引擎。

ADO 是微软开发的一套属于应用程序级的通用访问数据库编程接口,它提供一组优化的访问数据库专用对象集,是面向对象的数据库连接新技术,也为 ASP 提供了完整的站点数据库访问解决方案。ADO 可连接多种数据库管理系统,如 SQL Server、Oracle、Informix 等。ASP 与 ADO 结合,在服务器端脚本中,提供对数据库的操作,形成含有数据库信息的主页。通过嵌入 SQL 语句,使用户可以在浏览器端使用和管理数据库,通过浏览器页面输入、更新和删除服务器端的数据库的内容。

在 ASP 中内置了数据库访问组件 ADODB,它是属于数据库应用的 COM 构件,可以在多种环境下应用,ADO 通过它来访问各种类型的数据库。各种脚本和语言可以调用 ADO 组件访问数据库,并利用相应的数据接口显示查询结果。ADO 使用内置的 RecordSets 对象作为数据的主要接口,返回对数据库的查询结果。ADO 也可使用 VBScript、JavaScript 语言来控制对数据库的访问并显示查询的结果。

ASP 访问数据库的机制见图 1-9。

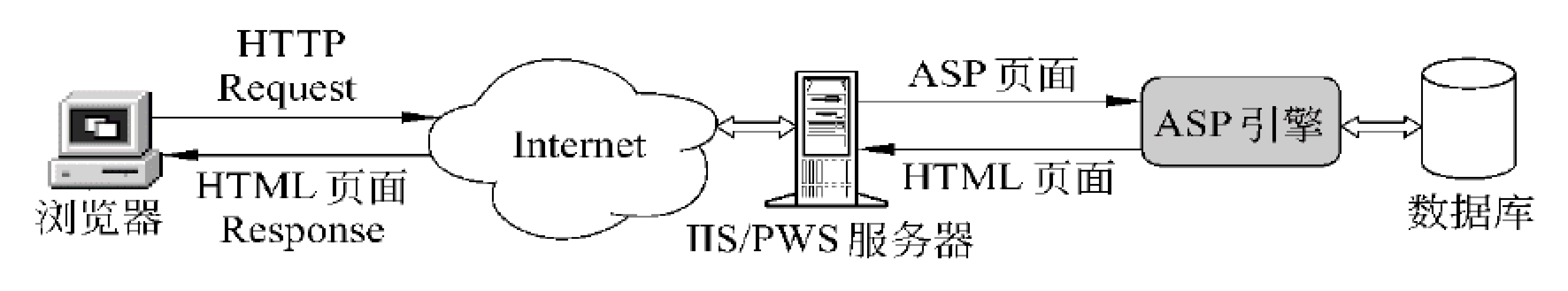


图 1-9　ASP 访问数据库的机制

ASP 的工作过程如下:

(1) 在浏览器地址栏输入要请求 *.asp 页面的 URL,发出一个 ASP 请求(Request)。

(2) IIS 服务器接受请求,根据扩展名.asp 识别出 ASP 文件,并找出相应的 ASP 文件。

(3) IIS 服务器把 ASP 文件发送到 ASP 引擎(asp.dll 动态连接库)。

(4) ASP 引擎将 ASP 文件从头至尾解释处理,调用相应的脚本引擎。若脚本指令调用了 ADO 组件,由 ADO 调用 ODBC,通过 ODBC 与后台数据库相连。

(5) 数据库管理系统对数据库进行操作,并将请求的数据通过数据库管理系统传到

ASP 引擎。

(6) ASP 引擎将执行结果动态生成一个 HTML 页面返回 IIS 服务器。

(7) IIS 服务器将 HTML 页面(Response)返回浏览器。

(8) 浏览器解释执行 HTML 页面,并显示结果。

3. PHP 技术

PHP 与 ASP 类似,也是一种服务器端的脚本运行环境,可以说它是 UNIX 系统上的 ASP。它与 ASP 的不同点在于:PHP 可以跨多个平台,在改变平台时,不需要修改 PHP 代码;PHP 支持数种 Web 服务器,如 IIS、Apache 等;对数据库的操作功能强大,优良的兼容性,使用 PHP 可以操作当前几乎所有的数据库,如 SQL Server、MySQL、Oracle、Solid 等。

PHP 能够把浏览器、OS 平台、Web 服务器和 Web 数据库技术融合在一起,形成一个完整理想的整体解决方案。PHP 混合了 C、Java、Perl 以及 PHP 式的新语法,在保证最大可操作性的前提下,提供了比一般 CGI 更快的速度。

PHP 的主要技术特点如下:

- PHP 具有良好的跨平台特性,支持数种主流 Web 服务器。
- PHP 属于开放资源软件,可以在 http://www.php.net 站点免费下载。
- PHP 简单易学。它也属于脚本语言,语法相当简单,不需要很强的语言基础。
- 数据库操作功能强大。它通过多种函数支持数种主流和非主流数据库。PHP 支持 ODBC 访问数据库技术。
- PHP 支持面向对象的概念,与 C++ 和 Java 类似。
- 使用 PHP 技术的最佳组合是:Linux、Apache 和 MySQL。

PHP 访问数据库的机制见图 1-10。

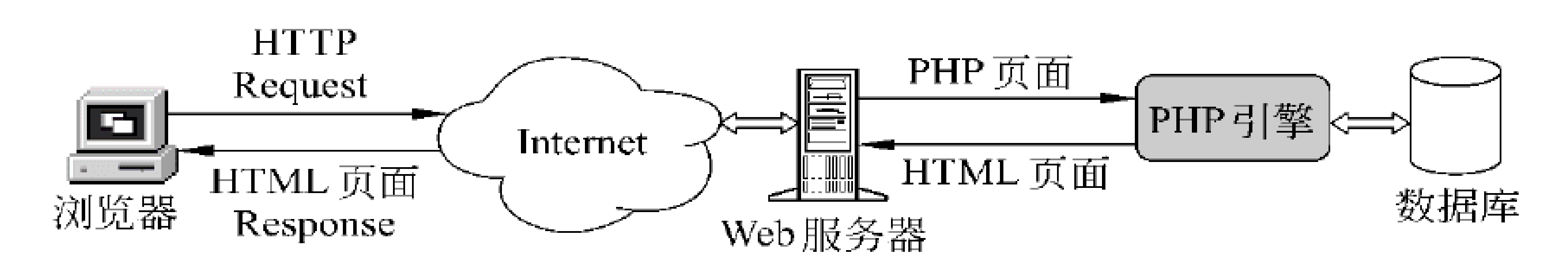


图 1-10 PHP 访问数据库的机制

PHP 的工作过程如下:

(1) 在浏览器地址栏输入要访问的 PHP 页面的 URL,发出一个 PHP 请求。

(2) Web 服务器接受该请求,根据扩展名.php 识别出 PHP 文件,并找出相应的 PHP 文件。

(3) Web 服务器把 PHP 文件发送到 PHP 引擎。

(4) PHP 引擎将 PHP 文件从头至尾进行扫描,根据命令连接后台数据库。

(5) 数据库管理系统对数据库进行操作,并将请求数据通过数据库管理系统上传到 PHP 引擎。

(6) PHP 引擎将执行结果动态生成一个 HTML 页面返回 Web 服务器。

（7）Web 服务器将 HTML 页面返回浏览器。

（8）浏览器解释执行 HTML 页面，并显示结果。

习题与实训 1

一、习题

1. 简述名词：计算机网络、局域网、广域网、城域网和因特网。局域网、广域网和城域网英文名称的缩写是什么？

2. Internet 提供哪些主要服务？

3. 请说明 Web 的三要素是哪些？

4. Internet 上的主机指什么？

5. Internet 中最重要的两个通信协议是什么？

6. WWW 的英文全称是什么？它的主要功能是什么？

7. ASP 与 ASP.NET 技术有什么特点？

8. 网络通信协议的三要素是什么？

9. OSI 体系结构采用了哪些层次？各层的功能是什么？

10. 试比较 TCP/IP 的层次结构与 OSI 结构。

11. OSI 协议中上层指哪些层？下层又指哪些层？

12. 简述 IP 地址的含义。

13. IP 地址采用什么表示法？

14. 可以为两台主机分配同一个 IP 地址吗？可以为同一个网络的两台主机分配同一个 IP 地址吗？可以为两个不同网络的两台主机分配相同的主机地址吗？

15. IP 地址分为哪几类？它们各自适用于什么情况？

16. 为什么要使用子网掩码？子网掩码的作用是什么？

17. Internet 上主机的 IP 地址和域名的关系是什么？

18. 以下几个 IP 地址中哪几个是合法的 IP 地址？为什么？

1）204.1546.21.106

2）123.232.87.0

3）175.146.87.175

4）202.96.0.97

5）204.258.0.96

19. 已知主机的 IP 地址和它的子网掩码，试确定该主机所在网络的类型、网络号、子网号和主机编号。

1）主机 IP 地址：200.196.0.134，子网掩码：255.255.255.192。

2）主机 IP 地址：160.200.21.87，子网掩码：255.255.248.0。

20. 有一个企业有 60 台主机，内含两个子网，一个子网有 40 台主机，另一个有 20 台主机，请为该单位分配 IP 地址。

21. 域名解析系统的作用是什么?

22. URL 的组成是什么? 写出其标准的结构形式,并简述各部分的功能。

23. 请解释 http://www.buu.edu.cn/wwwroot/default.html 的含义。

24. 在 Browser/Server/Database Server 三层工作模式中,Browser、Server、Database Server 分别起什么作用? 该模式有哪些优点?

25. Web 的体系结构是什么样的? 请简述它的工作过程。

26. 什么是 Web 数据库访问技术? 它的特点是什么?

二、实训课题

1. 请根据读者所在单位的实际情况,规划出所用主机的域名。

2. 已知某学院计算机布局如图 1-11 所示,请为该学院设计网络系统并规划网络 IP 地址。

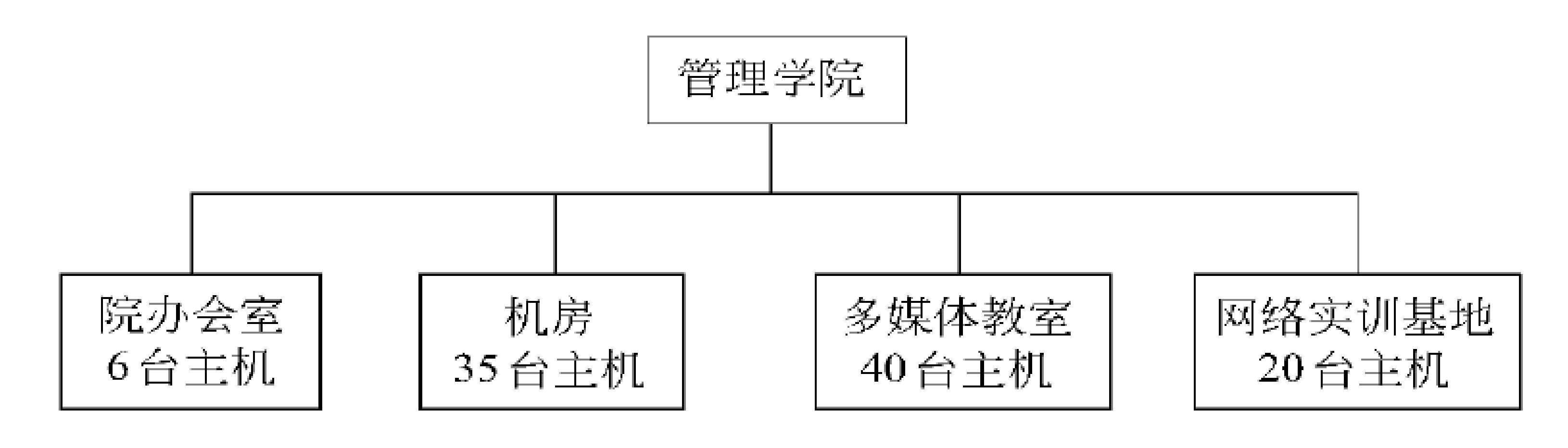


图 1-11　管理学院计算机布局图

第 2 章　Web 应用环境构建技术

Web 应用环境是 Web 应用开发的软硬件平台。一个实际 Web 应用环境往往需要较大的投资，作为一个学习过程，读者可以根据实际情况建立投资较小的仿真或实验环境，同样可以完成 Web 的开发工作。如果只有一台计算机，可以用一台机器虚拟一个网络环境；如果有两台计算机，可以用一台做服务器，另一台做客户机，模拟一个网络环境；如果已有机房，再配置一些网络设备，构成一个 Web 仿真运行环境，进行开发研究工作。通过本章学习，读者将了解各种常用的 Web 运行环境，并掌握 JSP 环境构建技术，为 Web 应用开发奠定基础。

2.1　Web 运行环境概述

2.1.1　园区内 Web 运行环境

在园区内建设一个 Web 应用环境，同时它也是一个小企业网络环境，读者可以在该环境完成 Web 站点的安装、配置、测试与应用开发工作。图 2-1 是一个小企业 Web 应用的运行环境，该环境具有一定的代表性，花费不多且易于实现，如果需要可以设置几个相同的组，供实验用。

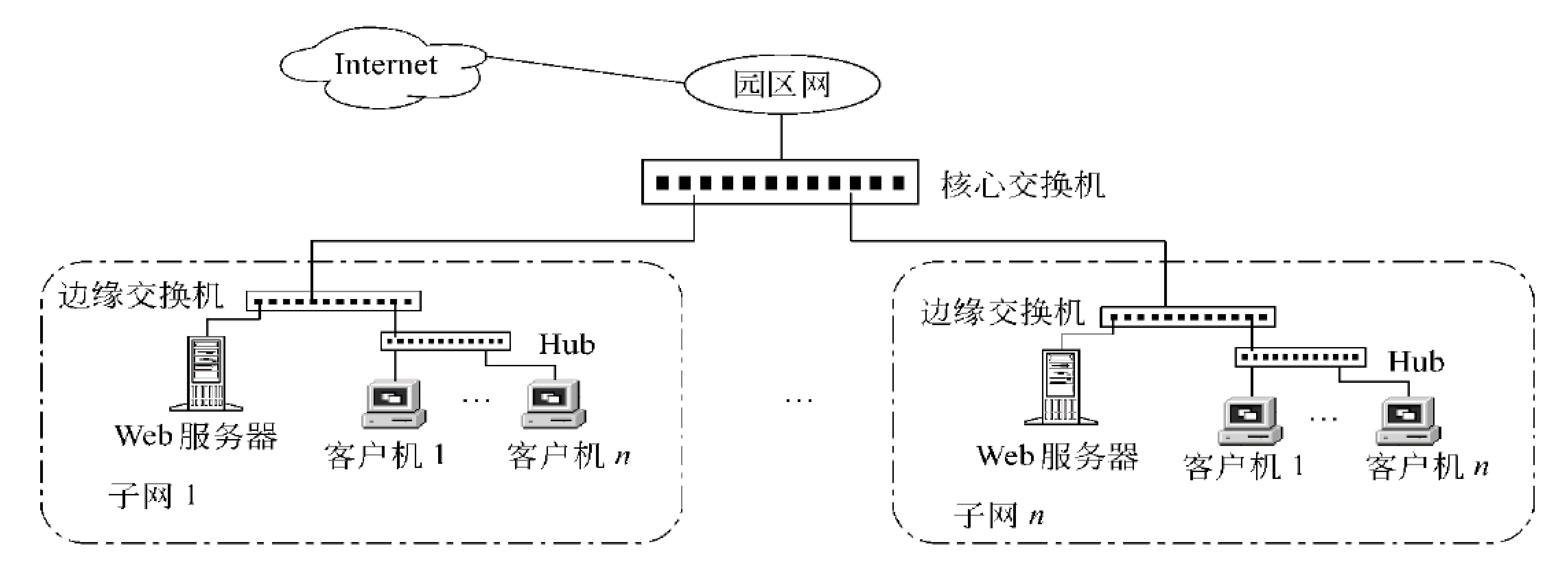


图 2-1　园区内的 Web 运行环境

图 2-1 示意了几个通过园区网接入 Internet 的子网。子网由一台网络服务器主机和若干台客户机组成,在服务器上安装一些基本的网络服务器软件,用以提供必要的网络服务,并实现对子网的管理。客户机上可以安装不同的操作系统,供开发使用。不同的子网可以安装不同的操作系统和基于该操作系统的服务器软件。读者既可以在这样一个环境中进行 Web 的各种集成开发工作,也可以正式对外发布信息。

2.1.2 模拟 Web 运行环境

图 2-2 是一个简单的模拟 Web 运行环境,用两台 PC 机联网,一台做 Web 服务器,一台做客户机,进行实验开发工作。

2.1.3 虚拟 Web 运行环境

如果只有一台计算机,可以虚拟一个 Web 运行环境,客户端和服务器均在一台计算机上,完成大多数 Web 的应用开发与测试工作,如图 2-3 所示。

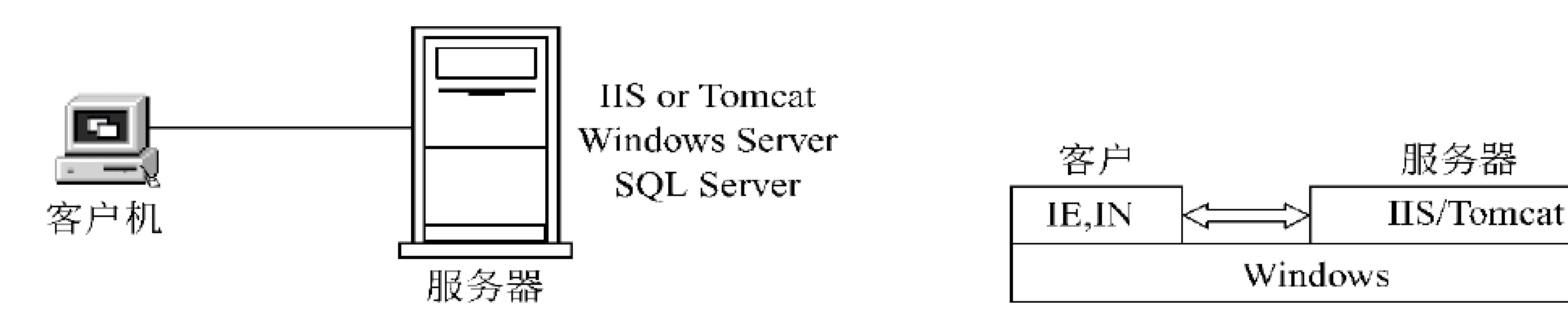


图 2-2 模拟 Web 运行环境　　图 2-3 虚拟 Web 运行环境

2.1.4 常用 Web 运行环境介绍

2.1.4.1 JSP 运行环境

JSP 运行环境下,可用的 Web 服务器软件有多种,如 resin、gnujsp、Servletexec、Tomcat 和 Jrun 等。以下提供了 3 种常用的运行环境。

1. 基于 Windows 2000/Windows XP 的 JSP 运行环境

- 服务器端操作系统:Windows 2000/Windows XP
- Java 开发工具:JDK
- 后台 Web 服务器:Tomcat
- 数据库:Access 2000/SQL 2000 个人版
- 客户端操作系统:Windows 2000/Windows XP
- 支持软件:IE 浏览器或其他兼容的浏览器
- 编程软件和开发工具:Dreamweaver、Flash、FrontPage、Eclipse 等

2. 基于 Windows 200X Server 的 JSP 运行环境

- 服务器端操作系统：Windows 200X Server
- Java 开发工具：JDK
- 后台 Web 服务器：Tomcat
- 数据库：SQL Server 企业版
- 客户端操作系统：Windows
- 支持软件：IE 浏览器或其他兼容的浏览器
- 编程软件和开发工具：Dreamweaver、Flash、FrontPage、Eclipse 等

3. 基于 Linux 的 JSP 运行环境

- 服务器端操作系统：Linux
- Java 开发工具：JDK
- 后台 Web 服务器：Tomcat 等
- 数据库：SQL Server 企业版
- 客户端操作系统：Windows
- 支持软件：IE 浏览器或其他兼容的浏览器
- 编程软件：Dreamweaver、Flash、FrontPage 等

2.1.4.2 ASP/ASP.NET 运行环境

在 Windows 平台上使用 ASP/ASP.NET 技术，一般采用 IIS（IIS Internet Information Server）作为 Web 服务器。IIS 是一个集成的 Web Server，它有创建、管理以及查找任何格式信息的一整套工具。IIS 提供管理和维护 Web 站点的能力，支持 WWW、FTP 和 gopher 服务。Windows 上 ASP/ASP.NET 的运行环境如下。

- 服务器端操作系统：Windows Professional/Server
- 后台 Web 服务器：IIS
- 数据库：SQL Server 或其他数据库
- 客户端操作系统：Windows
- 支持软件：IE 浏览器或其他兼容的浏览器
- 编程软件和开发工具：Dreamweaver、Flash、FrontPage、Eclipse 等
- ASP.NET 环境：Visual Studio.Net

2.1.4.3 PHP 运行环境

PHP 具有跨平台特性，可以与多种数据库连接，一般来讲 Linux＋Apache＋MySQL 是它的最佳组合。可以选用以下环境。

- 服务器端操作系统：Linux
- 后台 Web 服务器：Apache
- 数据库：MySQL

• 客户端操作系统：Windows
• 支持软件：IE 浏览器或其他兼容的浏览器
• 编程软件：Dreamweaver、Flash、FrontPage 等

Linux 是一套免费使用和自由传播的类 UNIX 操作系统，Linux 以它的高效性和灵活性著称。它能够在 PC 计算机上体现 UNIX 特性，具有多任务、多用户的能力。Apache 是基于 UNIX 平台的，它功能强大，安全系数高，而且十分稳定。

2.2 JSP 应用环境的建立

JSP 运行环境可以有多种选择，本书选用常用、易安装的运行环境，Windows XP 操作系统之上的 JDK、Tomcat 和 SQL 2000。

2.2.1 JSP 安装准备工作

安装 JSP 运行环境，需要准备以下软件：

（1）Java 的软件开发工具包（Java Development Kid，JDK）。本书使用版本 jdk 1.6.0，读者可以从 Java 公司网站 http://java.sun.com 免费下载到 jdk-6-windows-i586.exe 文件。

（2）支持 JSP（Servlet）的 Web 服务器。目前存在有多种 JSP Web 服务器软件，比较有名的有 Apache 的 Tomcat、Caucho.com 的 resin、Allaire 的 Jrun、New Atlanta 的 ServletExec、SUN 的 Java Web Server 和 IBM 的 WebSphere 等。本书选用 Tomcat，读者可以从 http://tomcat.apache.org 站点下载，文件名为 apache-tomcat-5.5.26.zip 或 apache-tomcat-5.5.26.exe。

（3）数据库管理系统 SQL Server 2000（Personal）。

JSP 是在 HTML 中嵌入 Java 代码的脚本。由 JSP 引擎先将其转化为 Servlet，然后再调用 Javac 将 Servlet 编译为 Class 文件，最后，由服务器解释执行 Class 文件。Servlet 是一个特殊的 Java 类，它一般从 HttpServlet 类继承而来，在这个类中要实现 doGet 或者 doPost 函数，由这两个函数处理来自客户的请求，并将结果返回。Servlet 和 JSP 是 Sun 公司 J2EE 架构中重要的部分。由于它基于 Java 语言，因此可以方便地调用功能强大的 Java API（如 JDBC）。

JDK 是通用的 Java 环境，可以在绝大多数计算机环境中运行，没有特殊的软硬件环境要求。安装前应退出所有正在运行的程序，然后执行安装程序。

2.2.2 安装配置 JDK

1. 安装 JDK

（1）在 Windows XP 下，直接双击 jdk-6-windows-i586.exe 文件，系统自动进入安装

进程。

（2）出现“许可证”对话框，选择接受协议，单击“下一步”按钮。

（3）出现“自定义安装”对话框，如果需要更改安装路径，单击“更改”按钮，改变安装路径，本书将 JDK 安装到 D 盘，采用路径为 D:\Java\jdk1.6.0\，见图 2-4，单击“下一步”按钮。

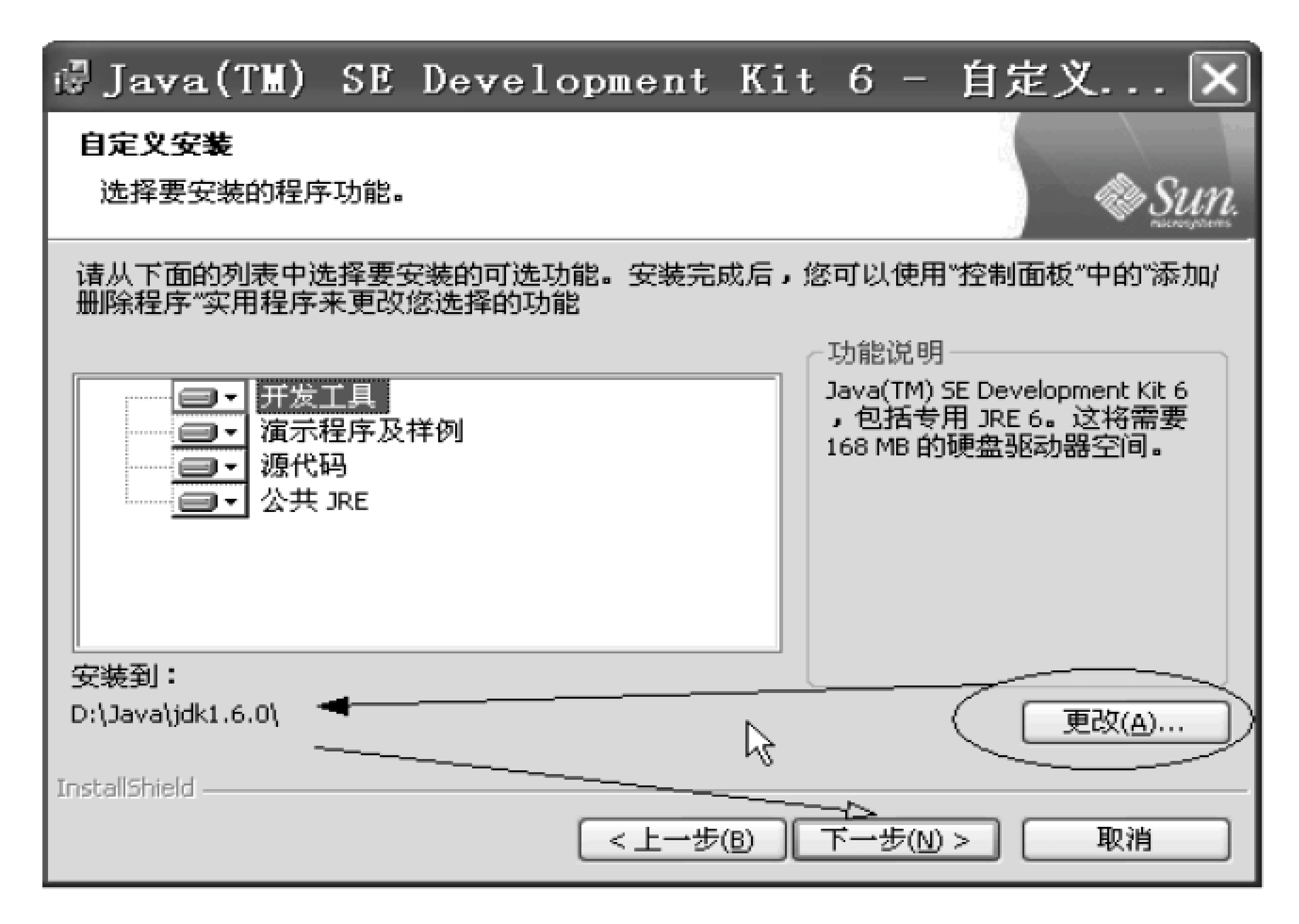


图 2-4　自定义安装对话框

（4）出现图 2-5 所示的对话框，选择需要安装的程序功能。本书需要更改安装路径，单击“更改”按钮，改变路径为 D:\Java\jdk1.6.0\，单击“下一步”按钮。

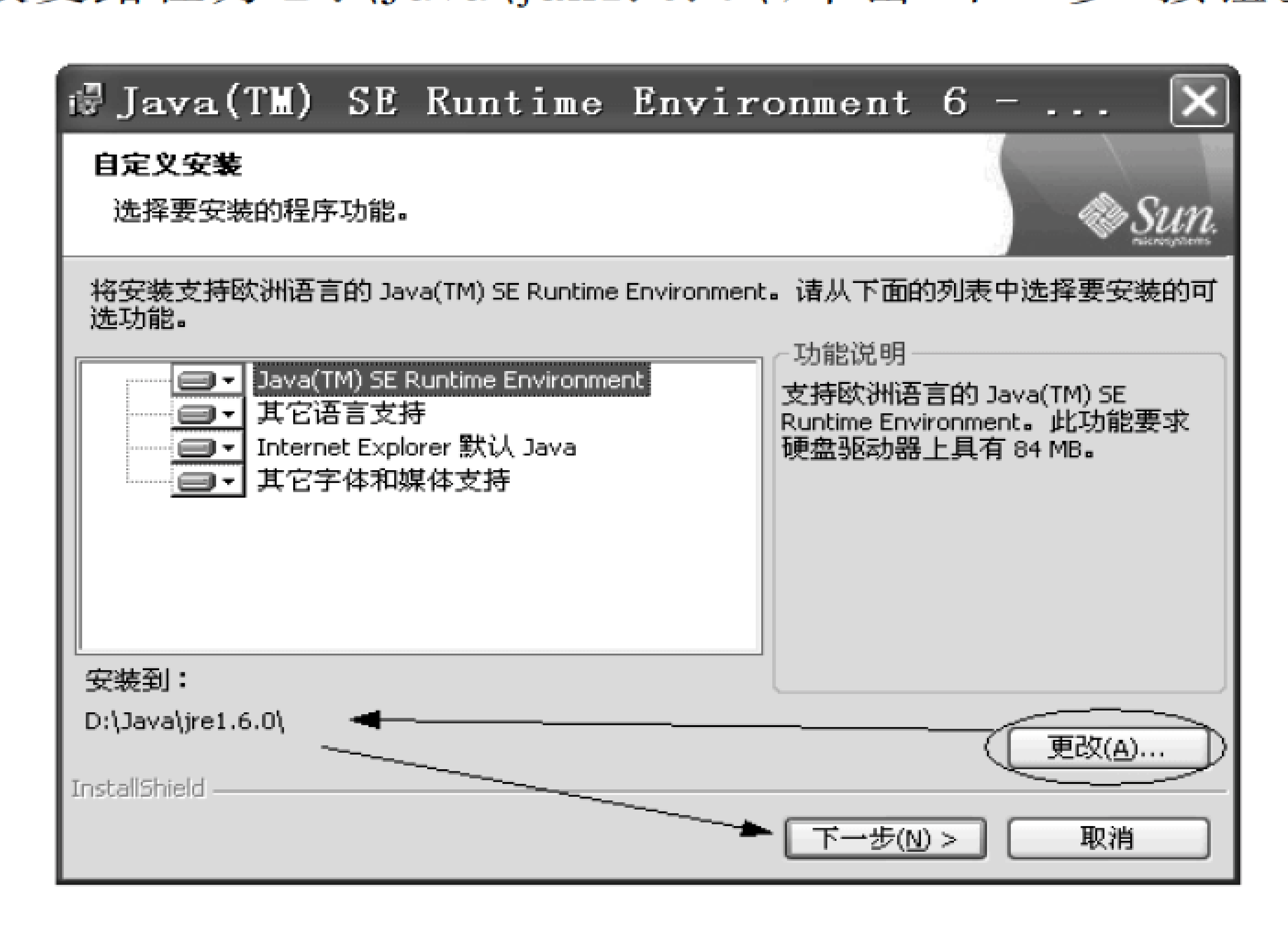


图 2-5　更改安装路径

（5）出现“安装完成”对话框，单击“完成”按钮，完成安装。

2. 配置系统环境变量

JDK 安装完成后应配置系统的环境变量，配置过程如下：

（1）在桌面上用鼠标右击“我的电脑”。

(2) 在下拉菜单中选择“属性”，弹出“系统属性”窗口，见图 2-6。在窗口中选择“高级”选项卡。

(3) 单击“环境变量”按钮。

(4) 出现“环境变量”对话框，如图 2-7 所示。在“系统变量”列表框中选择 Path 参数，并单击“编辑”按钮。



图 2-6　“系统属性”窗口



图 2-7　选择路径参数

(5) 出现“编辑系统变量”对话框，在变量值中加入 Java 的路径 d:\Java\jdk1.6.0\bin，然后单击“确定”按钮，如图 2-8 所示。

(6) 新建环境变量 JAVA_HOME。在图 2-7 的“系统变量”选项组中，单击“新建”，出现“新建系统变量”对话框，用类似方法设置环境变量 JAVA_HOME。环境变量 JAVA_HOME 的变量值是 d:\Java\jdk1.6.0。



图 2-8　加入 Java 路径

配置完成后，要重新启动计算机，环境变量才能生效。

3. 测试安装结果

(1) 用文本编辑器编写一个简单的 Java 程序：

```
public class HelloWorld {
public static void main(String args[])
{
 System.out.println("Hello World!"); }
}
```

这个例子就是著名的“Hello World!”,它的功能是显示“Hello World!”。

注意：该文件名必须命名为 HelloWorld.java,并区分大小写。

(2) 编译源文件,在 DOS 命令提示符下执行(注意区分大小写)：

```
Javac   HelloWorld.java
```

如果正常的话,在当前目录下生成 HelloWorld.class 文件。

(3) 运行,在 DOS 命令提示符下执行(注意区分大小写)：

```
java   HelloWorld
```

显示“Hello World!”,程序运行成功,说明已经成功配置好 JDK。

2.2.3 安装服务器软件 Tomcat

Tomcat 是一个 JSP 和 Servlet 的运行平台,是 Sun 公司在 JSWDK(Java Server Web Development Kit)基础上发展起来的优秀 Server/JSP Web 服务器。它可以和大部分主流 Web 服务器(如 IIS、Apache 等)一起工作,并且运行稳定、可靠、效率高。

Tomcat 服务器除了能够运行 Servlet 和 JSP 外,还提供了作为 Web 服务器的一些特有功能,如 Tomcat 管理和控制台、安全域管理等。有关 Tomcat 的信息,可以访问 Tomcat 主页：http://jakarta.oapache.org/tomcat/index.html。

1. Tomcat 安装

(1) 运行 apache-tomcat-5.5.26.exe 文件,出现“安装向导”对话框,在对话框中单击 Next 按钮。出现协议窗口,单击 I Agree 按钮,接受协议。

(2) 进入选择安装方式对话框,如图 2-9 所示,选择 Full 完全安装,单击 Next 按钮。

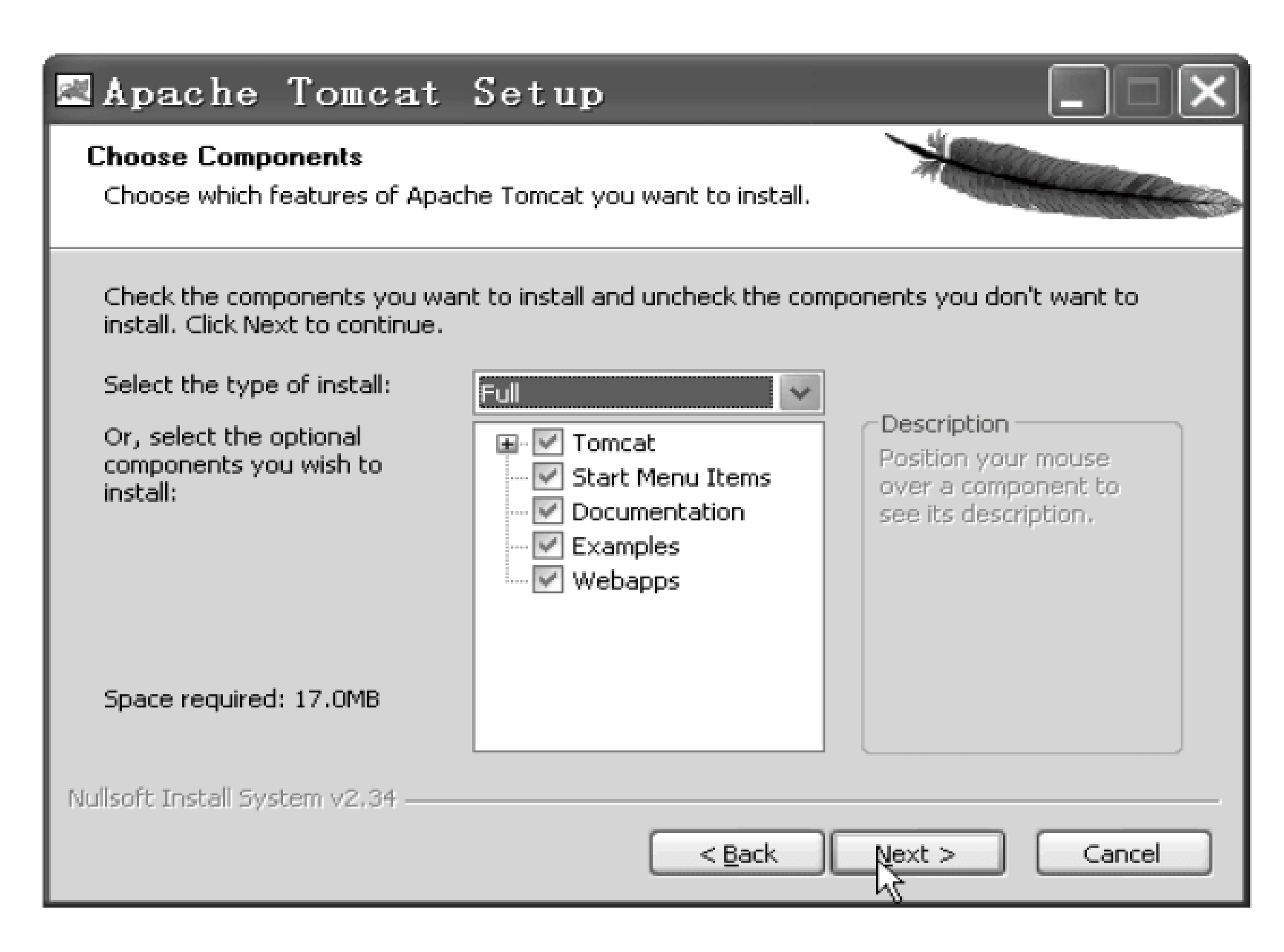


图 2-9　选择完全安装

(3) 出现选择安装目录对话框,见图 2-10,本书选择 d:\Tomcat5.5,单击 Next 按钮。

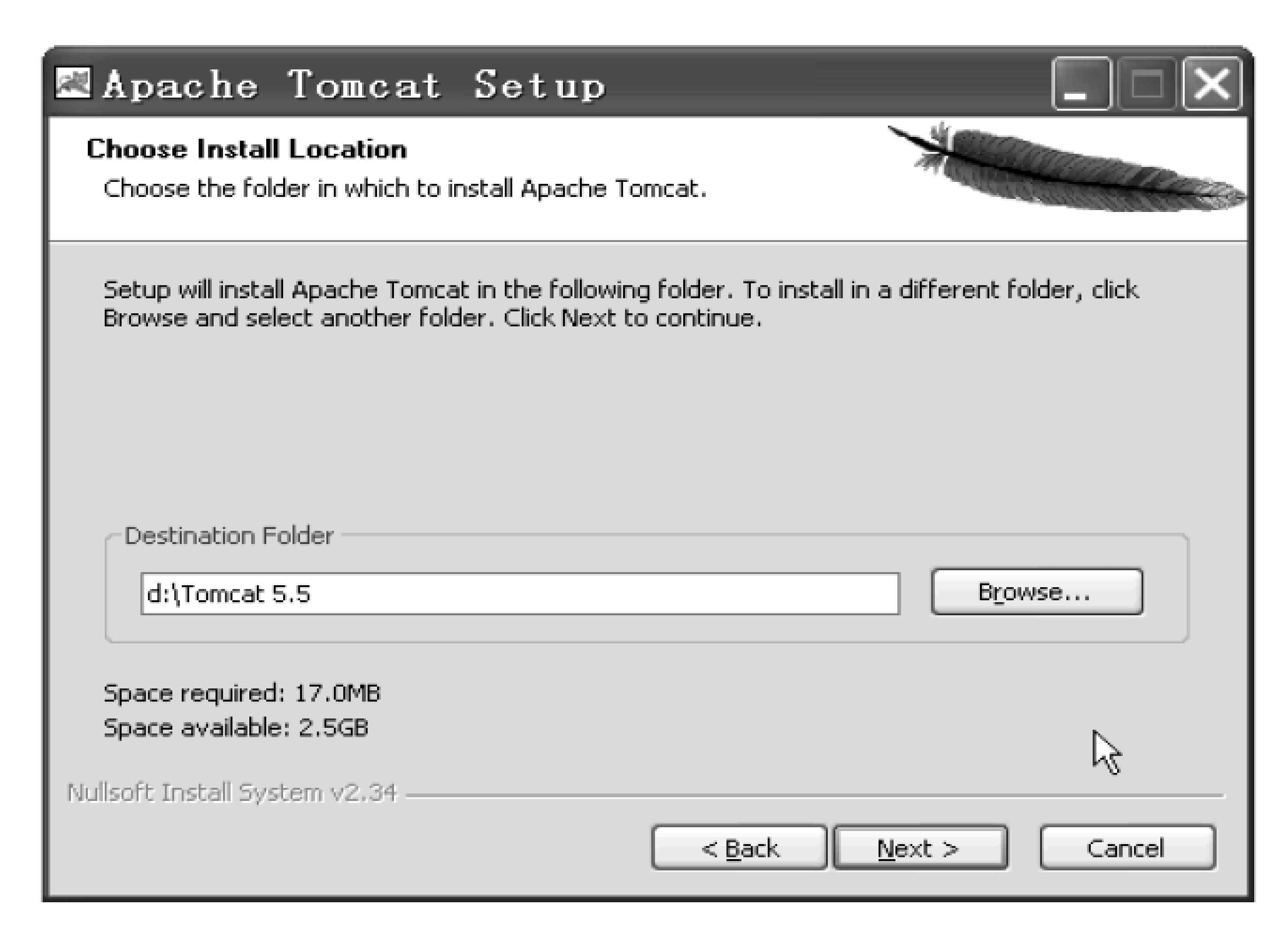


图 2-10　选择安装目录

（4）出现 Tomcat 基本配置对话框，如图 2-11 所示。Tomcat 默认 HTTP 的连接端口号为 8080，取默认值，用户名为 Admin，口令空（也可以设密码便于管理），单击 Next 按钮。

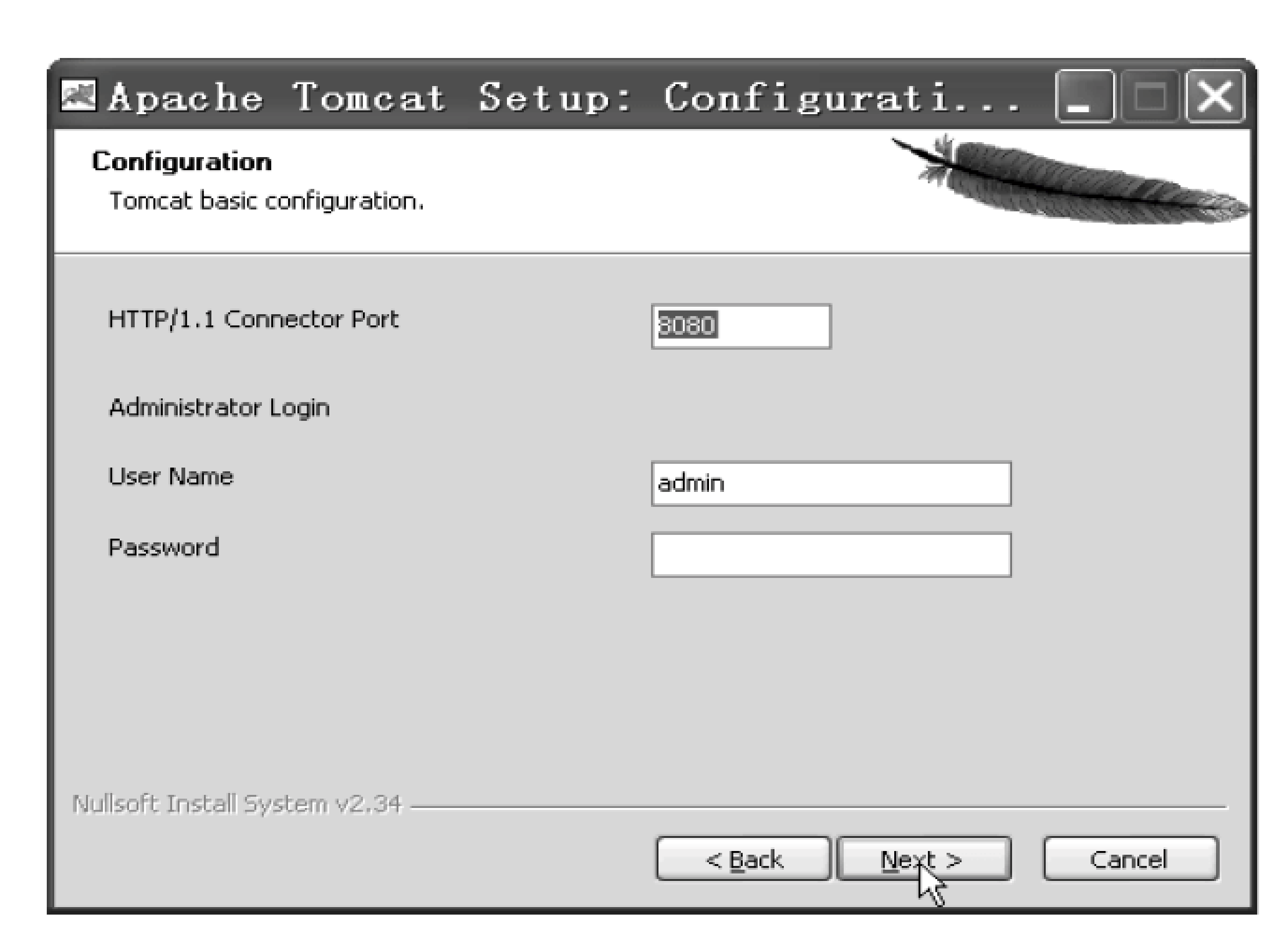


图 2-11　设置端口号和管理员信息配置

（5）出现选择 Java 虚拟机路径对话框，如图 2-12 所示，单击浏览按钮，找到 JDK 安装路径 D:\Java\jdk1.6.0\，单击 Install 按钮，系统将自动进行安装，直至结束。

2. 测试 Tomcat

安装完成后，在浏览器地址栏输入 http://localhost:8080 或 http://127.0.0.1:8080，可以看到 Tomcat 的默认主页，如图 2-13 所示，表示安装成功。

注意： Tomcat 默认的 Web 的连接端口号为 8080。使用浏览器浏览 Tomcat 主页时应说明此端口号。如果通过本机浏览，可在 URL 域键入：http://localhost:8080 或 http://127.0.0.1:8080，如图 2-13 所示。

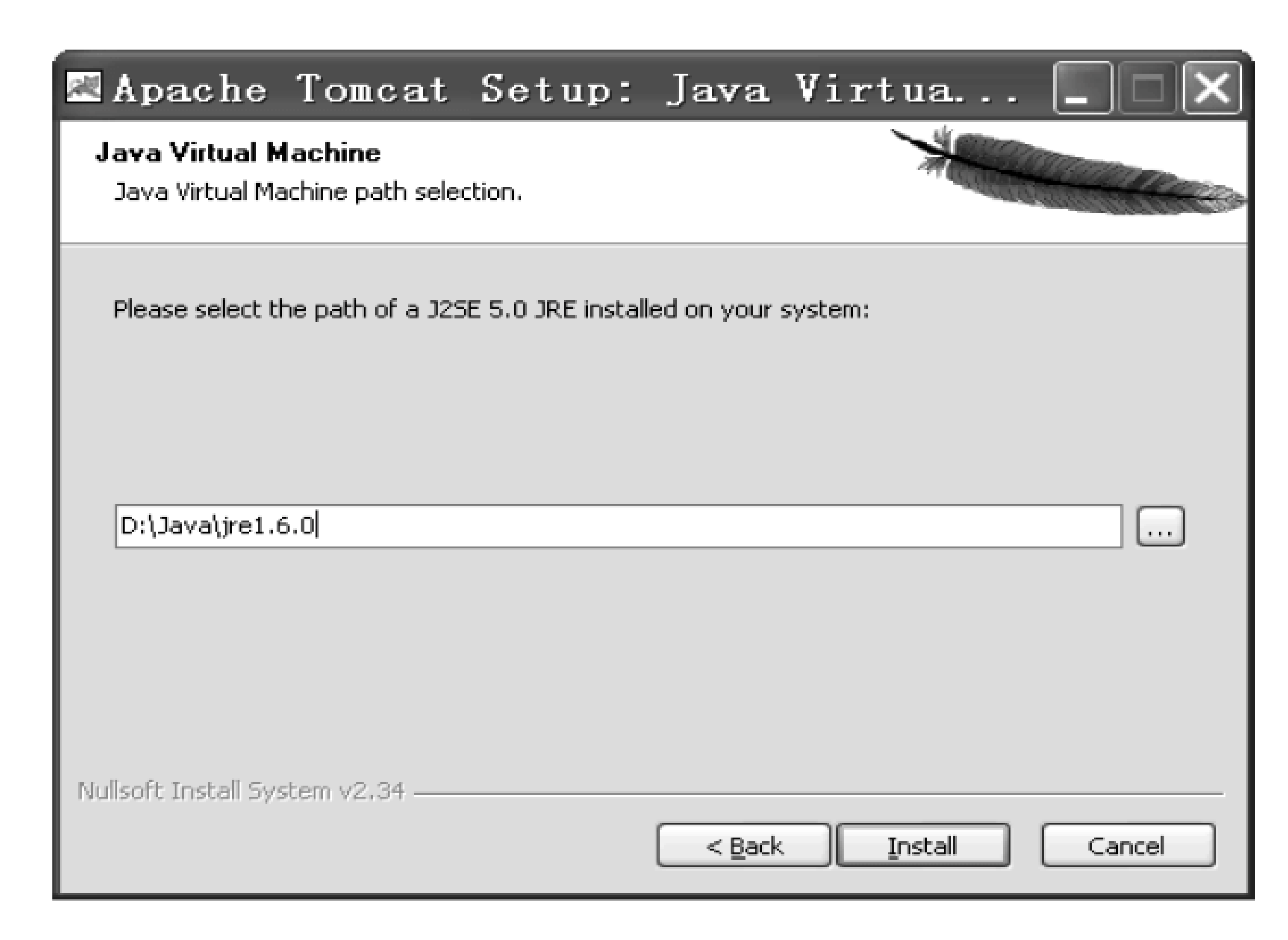


图 2-12　Java 虚拟机目录

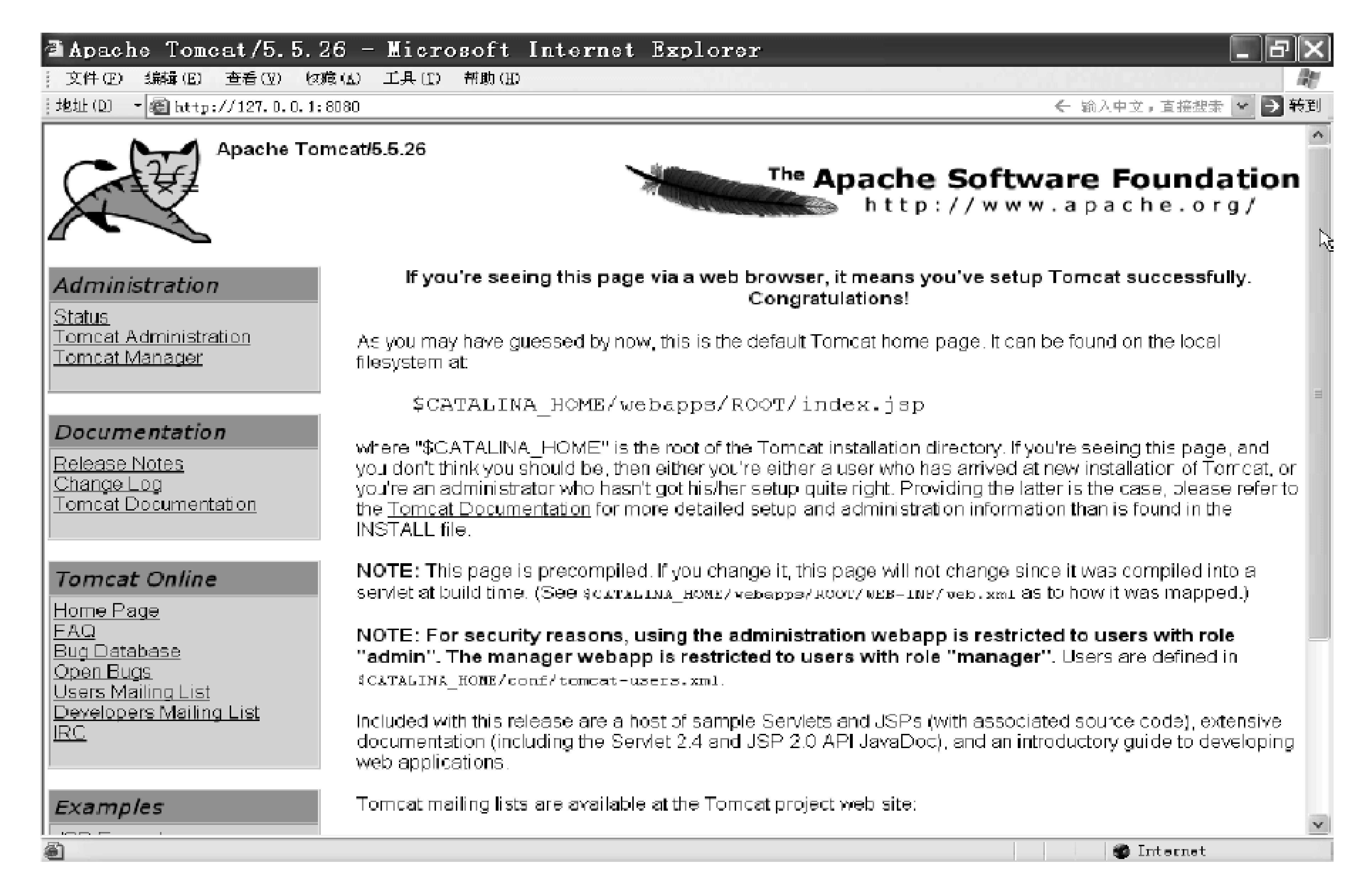


图 2-13　Tomcat 默认主页

3. Tomcat 的目录结构

安装完成后，目录结构如图 2-14 所示，根目录为 Tomcat 5.5。本书今后使用的 Tomcat 服务器，安装在 D 盘 Tomcat 5.5 目录下。

Tomcat 的目录结构参见表 2-1。

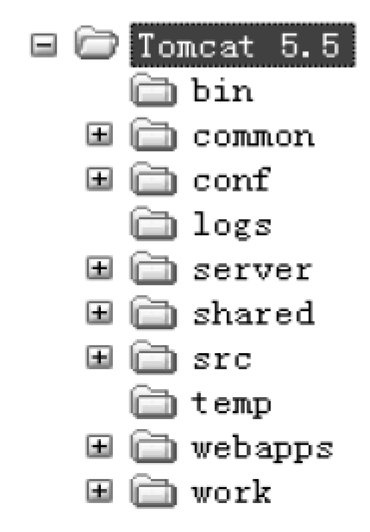


图 2-14　Tomcat 5.5 目录结构

4. Tomcat 的启动和停止

可以用两种方法启动与停止 Tomcat 服务器。

表 2-1 Tomcat 的目录结构

目 录	描 述
/bin	存放启动和停止 Tomcat 服务器的脚本文件
/conf	Tomcat 的配置文件,其中重要的配置文件是 server. xml
/server	包含 classes、lib 和 webapps 子目录
/server/lib	存放 Tomcat 服务器需要的 JAR 文件
/server/webapps	存放 Tomcat 自带的两个 Web 应用:admin 和 manager
/common/lib	存放 Tomcat 服务器及所有 Web 应用都可以访问 JAR 的文件
/shared/lib	存放所有 Web 应用都可以访问 JAR 的文件
/webapps	默认发布目录,Web 应用程序存放的地点
/logs	存放 Tomcat 的日志文件
/work	存放由 JSP 生成的 Servlet

(1) 安装完成或系统重新启动后,屏幕的右下角任务栏上出现 Tomcat 作业图标,见图 2-15(a)。如果没有该图标,通过 Windows 的"开始→所有程序→Apache Tomcat 5. 5→Monitor Tomcat",使 Tomcat 作业图标出现。Tomcat 启动后,作业图标上出现绿色箭头,停止后出现红色方块。鼠标右键单击该图标,出现菜单(见图 2-15(b)),在 Tomcat 启动状态下,选择 Stop service,停止 Tomcat 服务器。在 Tomcat 停止状态下,选择 Start service,启动 Tomcat 服务器。

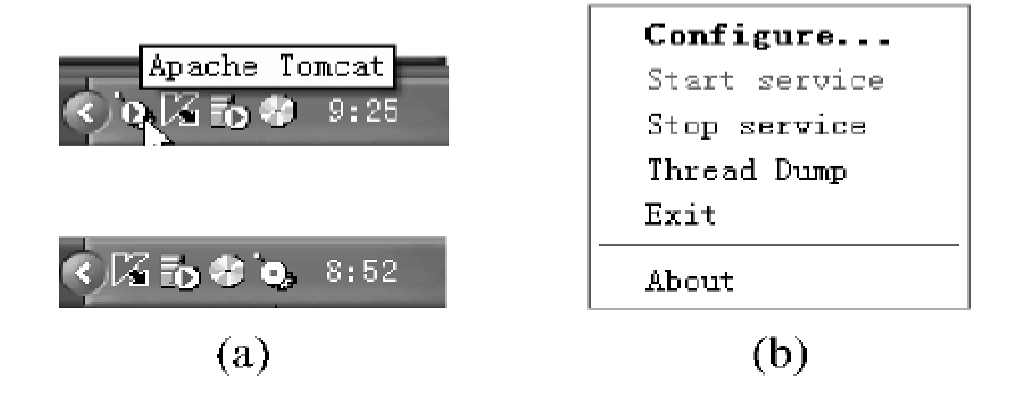


图 2-15 Tomcat 的启动与停止

(2) 运行 bin 目录下的 tomcat5. exe,启动 Tomcat。出现 DOS 窗口,显示批处理文件执行过程和 Tomcat 启动时间,见图 2-16。关闭 Tomcat 的 DOS 窗口,则停止 Tomcat 服务器。

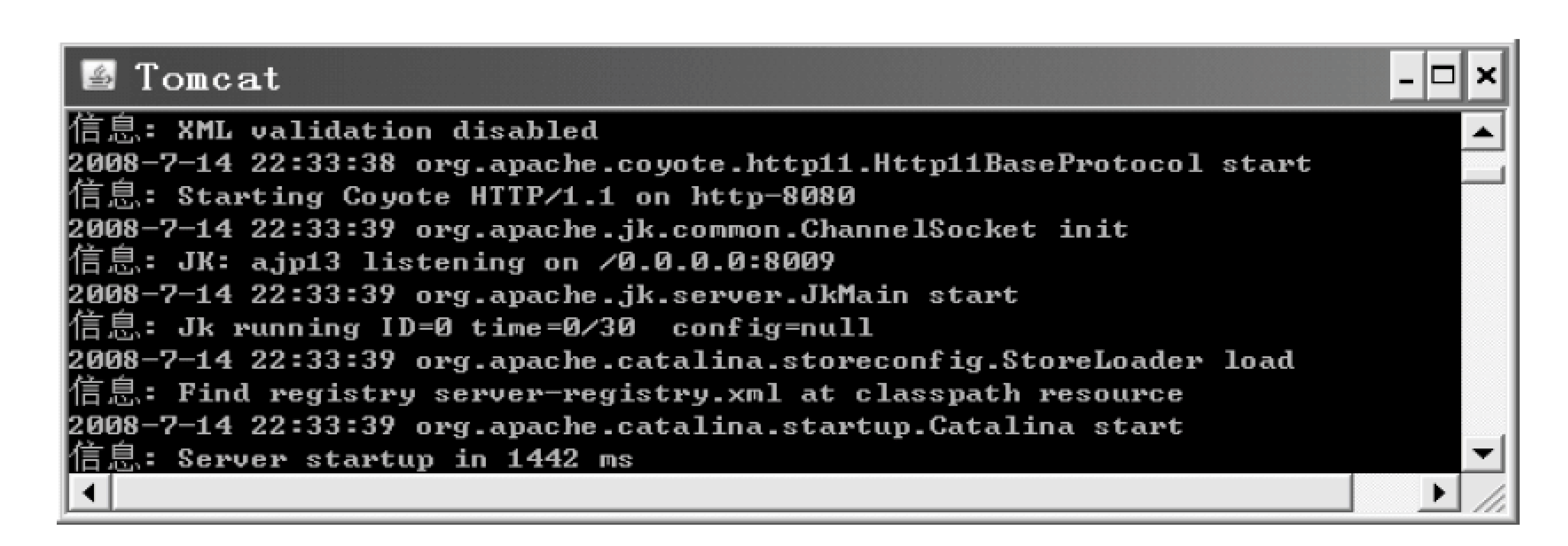


图 2-16 启动 Tomcat

注意:运行 bin 目录下的 tomcat5. exe,启动 Tomcat,将占用一个 DOS 窗口。使用作业图标启动/停止 Tomcat 服务器,可以不占用 DOS 窗口。

5. 创建虚拟目录

Tomcat 安装完成后，在 Tomcat 目录下已经生成了 webapps 发布目录。在 webapps 下的任何一个子目录，都是一个 Web 服务目录。

为使用方便，可以创建虚拟发布目录。虚拟目录是除了主发布目录以外的其他发布目录。虚拟目录在物理上可以不被包含在主目录中，但是逻辑上就像在主目录中一样。创建方法如下：在 Tomcat 的 conf 目录中，找到 server.xml 文件，在文件的最后找到 </Host>标记，在</Host>前添加以下语句：

```
<Context path="/BookShop" docBase=" d:\ex_D02" debug="0" reloadble="true"/>
```

设置完成后，重新启动 Tomcat，虚拟路径起作用。该语句将 D 盘上的 ex_D02 目录设为虚拟发布路径，它的别名为 BookShop。在浏览器的地址栏目输入 http://127.0.0.1:8080/BookShop/ex2-01.jsp，即可发布 d:\ ex_D02\ex2-01.jsp 文件。

（1）path 的值：虚拟目录的别名，在浏览器地址栏目中输入的路径。

（2）docBase 的值：是应用程序的实际路径，在硬盘上的存放位置，即绝对路径。

2.2.4 第一个 JSP 应用

（1）在记事本中输入以下代码，并存放在 Tomcat 的 webapps\ex_D02 目录下，文件名为：ex2-01.jsp。

```
<%@ page contentType="text/html; charset=GB2312"%>
<html>
<head><title>JSP 测试页面</title></head>
<body><center><font size=5 color=red>
<%="JSP 测试页面"%></font>
</center></body></html>
```

（2）在浏览器地址栏目输入 http://127.0.0.1:8080/ex_D02/ex2-01.jsp，代码运行结果如图 2-17 所示。



图 2-17 JSP 测试结果

2.3 SQL Server 数据库系统的安装与测试

数据库系统是开发动态 Web 页面不可缺少的系统支持环境。数据库系统在 Web 站点中的位置如图 2-18 所示。

SQL Server 2000 数据库系统是微软公司开发的数据库系统，SQL Server 2000 可以兼顾小、中、大规模的应用的需求，是一个与 Windows 操作系统结合的很好的数据库系统。

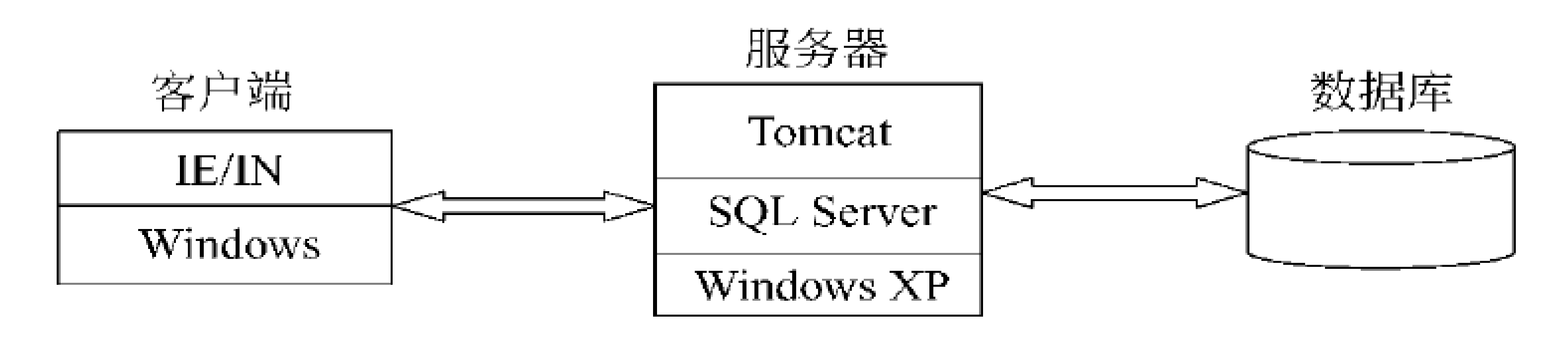


图 2-18　数据库系统

2.3.1　SQL Server 安装准备工作

(1) 使用具有本地管理权限的用户账户(Administrator)登录到操作系统。

(2) 关闭所有和 SQL Server 相关的服务。包括所有使用 ODBC 的服务，如 Microsoft Internet Information 服务 (IIS)。

2.3.2　安装 SQL

(1) 插入 SQL Server 2000 Personal Edition 光盘，自动进入安装界面。

(2) 单击“安装 SQL Server 2000 组件”，开始安装。系统要求输入安装 SQL 数据库的计算机的名称，如图 2-19 所示。



图 2-19　输入计算机名称窗口

(3) 选择“本地计算机”，并单击“下一步”，弹出“安装选择”窗口，如图 2-20 所示，继续安装。

(4) 在“安装选择”窗口中选择“创建新的 SQL Server 实例，或安装‘客户端工具’”单选钮。初次安装时应选择这一安装模式，不需要使用“高级选项” 进行安装。“高级选项”中的内容均可在安装完成后进行调整。选择完成后单击“下一步”，弹出“用户信息”窗口，如图 2-21 所示。

(5) 在“用户信息”窗口，单击“下一步”按钮继续安装，进入“安装定义”窗口，如图 2-22 所示。



图 2-20 “安装选择”窗口



图 2-21 “用户信息”窗口



图 2-22 “安装定义”窗口

(6) 选择“服务器和客户端工具”选项进行安装。需要将服务器和客户端同时安装，这样在同一台机器上，可以完成相关的所有操作，对于学习 SQL Server 很有用处。如果已经在其他机器上安装了 SQL Server，则可以只安装客户端工具，用于对其他机器上

SQL Server 的存取。单击"下一步"按钮，进入"实例名"窗口，如图 2-23 所示。

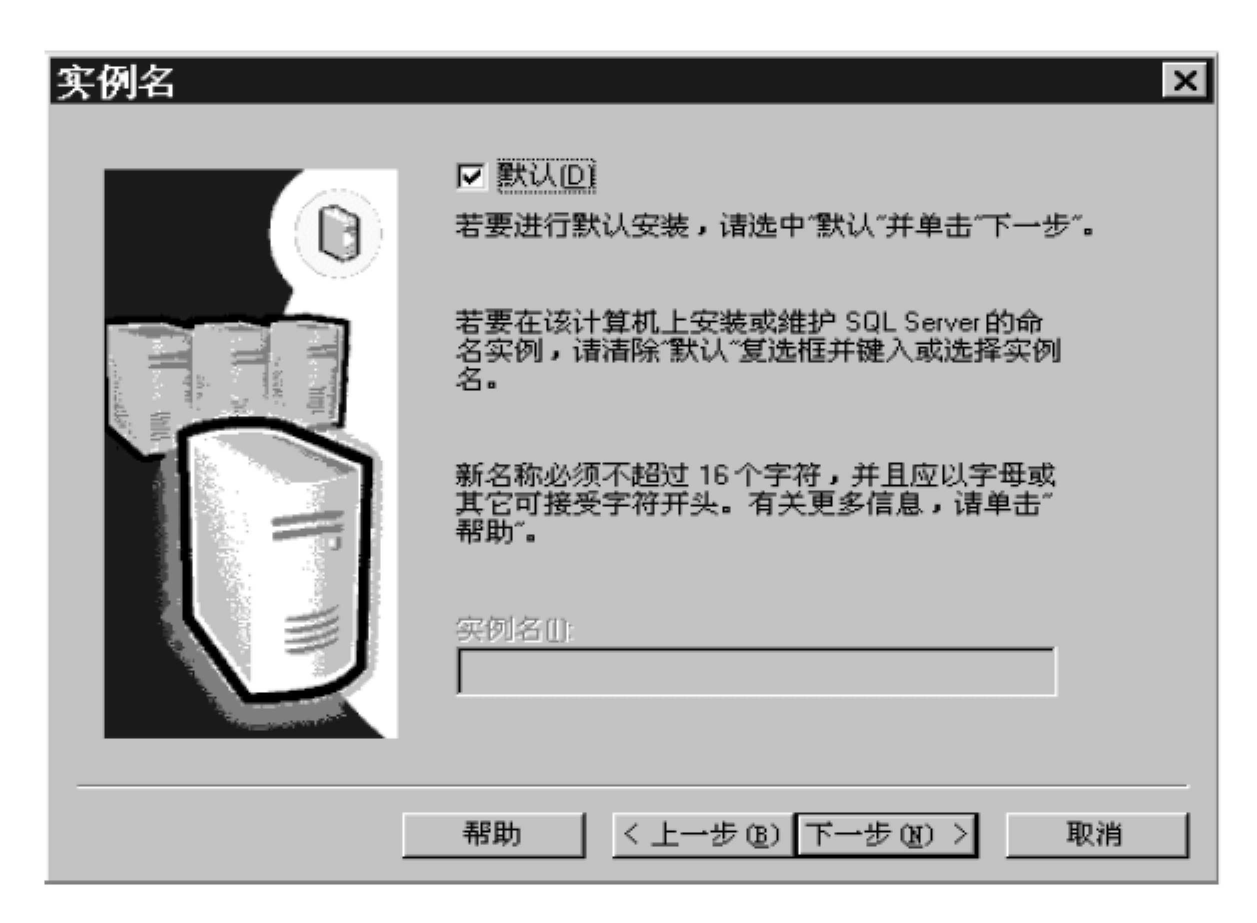


图 2-23 "实例名"窗口

(7) 在"实例名"窗口选择"默认"选项。单击"下一步"按钮，进入"安装类型"窗口，如图 2-24 所示。

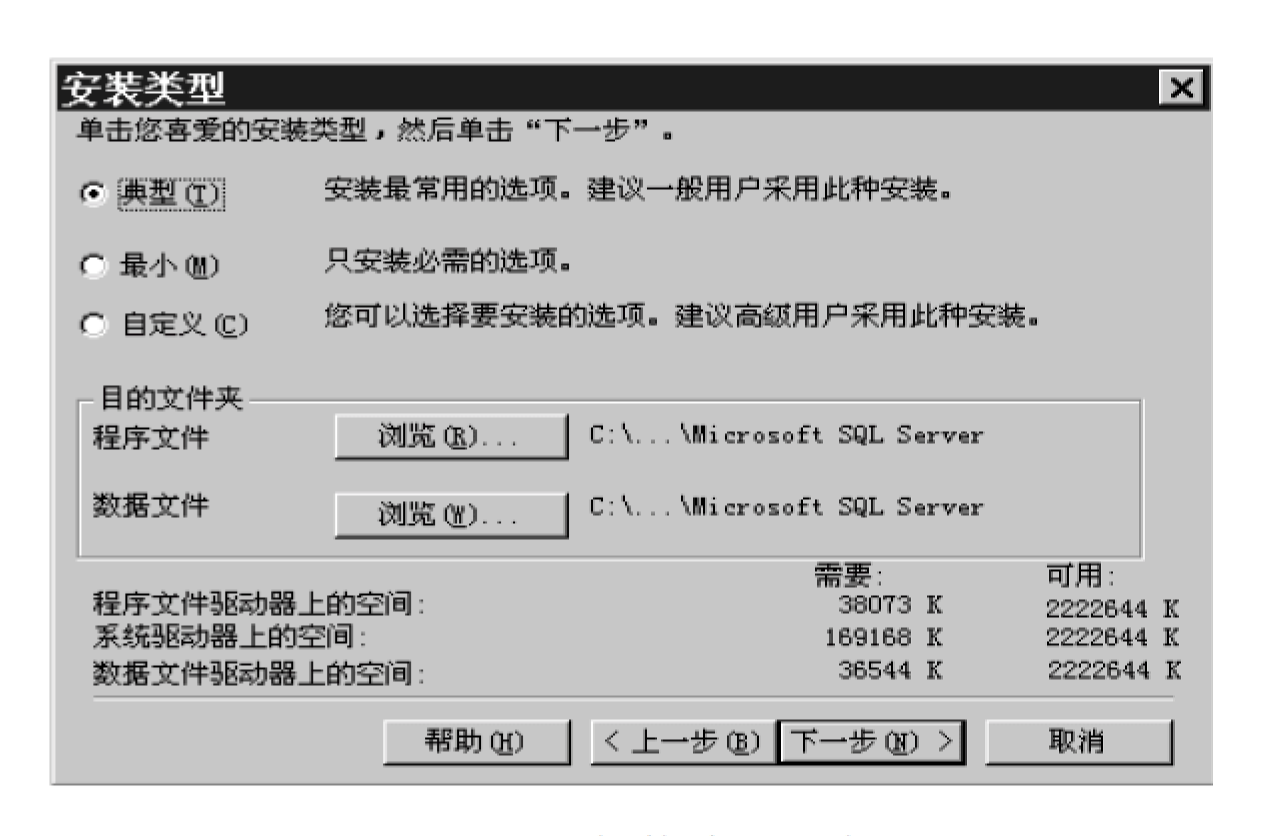


图 2-24 "安装类型"窗口

(8) 在"安装类型"窗口选择"典型"安装选项，并指定"目的文件夹"。程序和数据文件的默认安装位置都是 C:\Program Files\Microsoft SQL Server\。如果将 C 盘作为系统区、D 盘作为应用区使用，也可以选择 D 盘。注意，如果数据库数据有 10 万条以上的话，请预留至少 1GB 的存储空间，以应付需求庞大的日志空间和索引空间。单击"下一步"，进入"服务账户"窗口，如图 2-25 所示。

(9) 在"服务账户"窗口选择"对每个服务使用同一账户。自动启动 SQL Server 服务"选项。在"服务设置"处，选择"使用本地系统账户"，SQL Server 2000 将使用 Windows 2000 的用户管理体系。如果需要"使用域用户账户"的话，请将该用户添加至 Windows Server 的本机管理员组中。单击"下一步"按钮，进入"身份验证模式"窗口，如图 2-26 所示。

(10) 在"身份验证模式"窗口，选择"混合模式"选项，并设置管理员 sa 账号的密码，sa 账号是 SQL 数据库的数据库管理员的账号，如果输入了密码，请牢记密码。如果只是



图 2-25 “服务账户”输入窗口

身份验证模式

选择身份验证模式。

Windows 身份验证模式(W)

混合模式(Windows 身份验证和 SQL Server 身份验证)(M)

添加 sa 登录密码:

输入密码(E):

确认密码(F):

空密码(不推荐)(K)

帮助 < 上一步(B) 下一步(N) > 取消

图 2-26 “身份验证模式”窗口

为了学习的话,可以将该密码设置为空,以方便登录。如果是真正的应用系统,则千万需要设置和保管好该密码!如果需要更高的安全性,则可以选择“Windows 身份验证模式”,这时就只有 Windows Server 的本地用户和域用户才能使用 SQL Server 了。

(11) 输入以上信息后,系统已经得到了全部用户安装要求。系统将自动完成安装工作,直至“安装完毕”窗口出现,如图 2-27 所示,单击“完成”按钮结束安装。

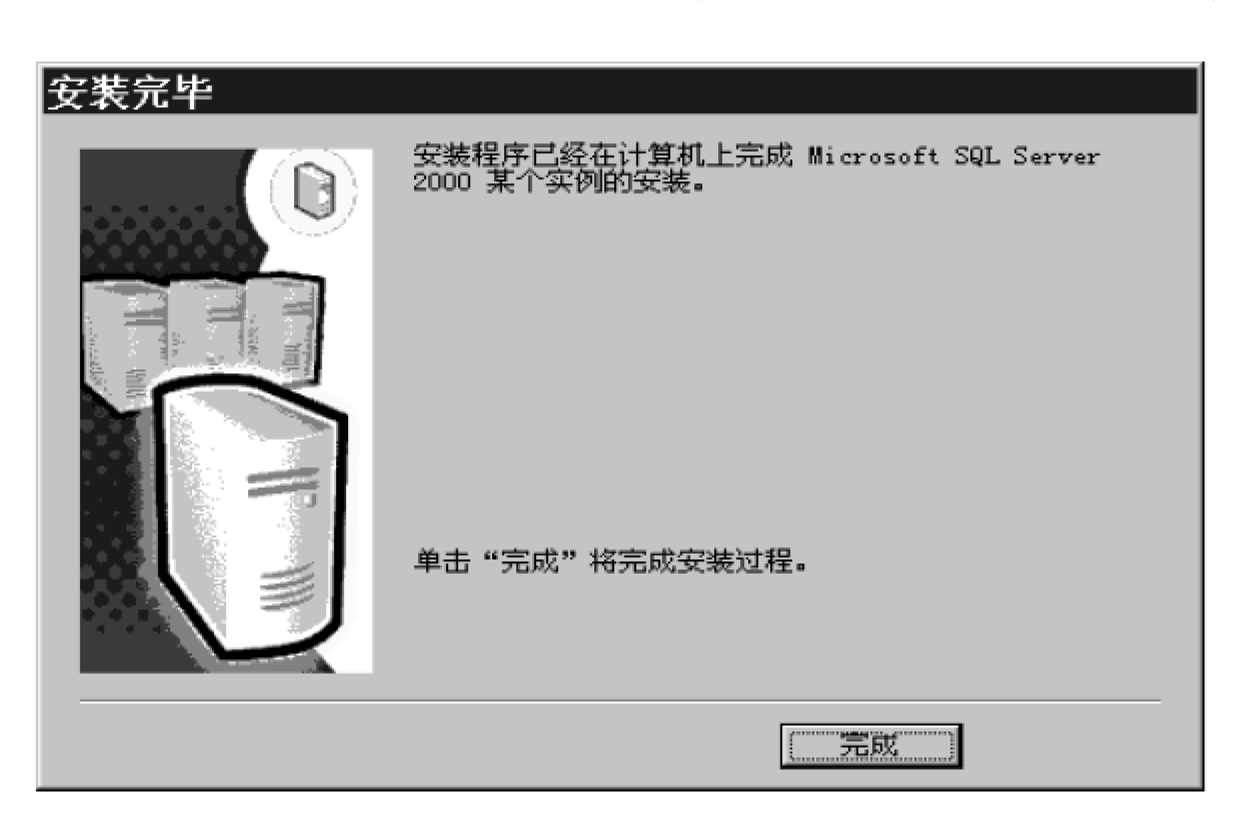


图 2-27 安装完毕窗口

2.3.3 SQL Server 的操作

1. SQL Server 2000 的启动和停止

在默认安装的情况下，SQL Server 2000 数据库将在 Windows 启动时自动启动。在 Windows 桌面的任务栏中显示 SQL 运行的图标，见图 2-28(a)。双击此图标打开 SQL Server 服务管理窗口，见图 2-28(b)。通过服务管理窗口，可以很方便地启动/停止、暂停/继续 SQL 数据库系统的运行，也可以定义 SQL 数据库是否在操作系统启动时同时启动。如图 2-28 所示。

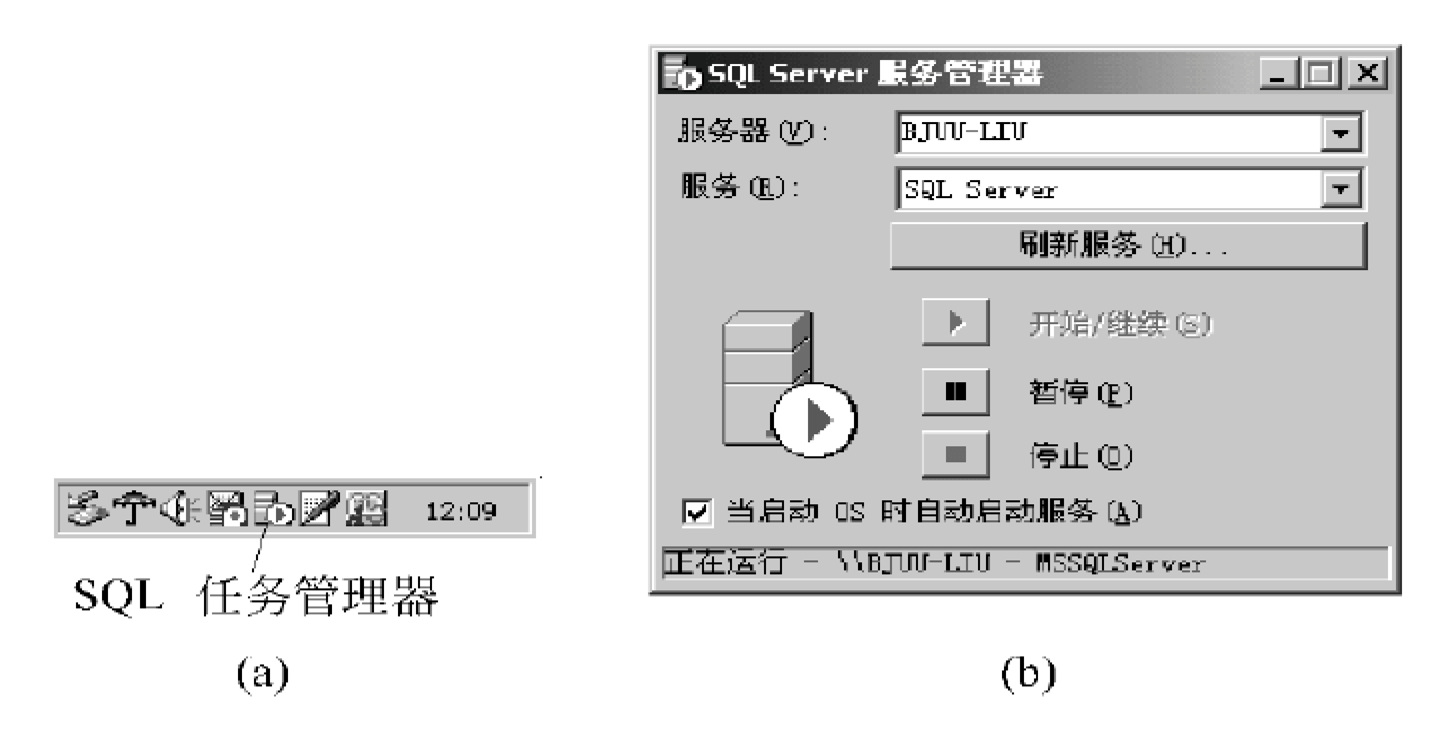


图 2-28　SQL Server 的任务管理

2. SQL Server 的管理工具

SQL Server 2000 系统提供了丰富的使用工具。数据库安装后，可以通过菜单方式方便地使用这些工具。

单击：开始→程序，找到 Microsoft SQL Server，系统将列出 SQL 的各种服务工具。如图 2-29 所示。

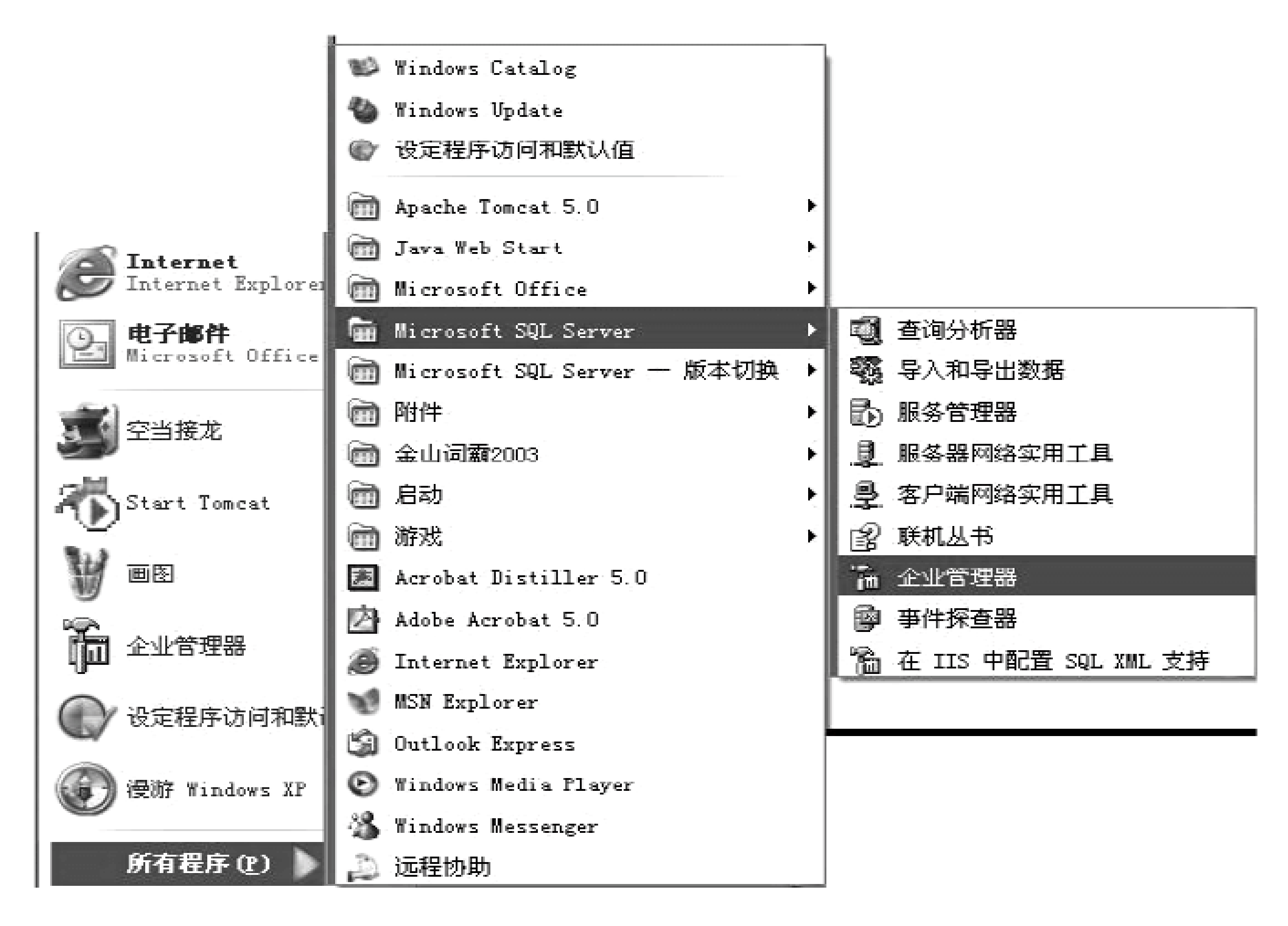


图 2-29　Microsoft SQL Server 管理工具

3. 查询分析器

查询分析器的主要用途是编辑 Transact SQL，将 Transact SQL 语句发送到服务器并显示查询后的结果。查询分析器中有两个窗口：上面区域输入 Transact SQL 语句；下面窗口显示执行结果。查询分析器是在数据库应用开发中使用频繁的工具。

4. 导入和导出数据

“导入和导出数据”是一个向导功能，它的功能是实现数据的传递，它不仅可以在 SQL 服务器间传递数据，而且可以在异种数据库间传递数据。

5. 服务管理器

服务管理器用于管理 SQL Server 启动/停止、暂停/继续以及是否在操作系统启动时自动启动等操作。

6. 服务器网络实用工具

服务器网络实用工具和客户端网络实用工具是提供 SQL 数据库网络连接的工具。服务器网络实用工具用于设置本机作为服务器时允许的网络连接协议，以支持不同配置的客户端。

7. 客户端网络实用工具

客户端网络实用工具用于本机作为 SQL 客户机访问其他 SQL 服务器时所使用的默认网络连接库，并可为不支持默认网络连接库的 SQL 服务器设置连接方式。

8. 联机丛书

联机丛书是一本联机文档，供联机查阅使用。它包含了 SQL Server 管理和开发的全部信息。

9. 企业管理器

SQL Server 企业管理器是 Microsoft SQL Server 2000 的主要管理工具，它提供了一个遵从 Microsoft 管理控制台（MMC）的用户界面，通过企业管理器可以完成以下操作：

- 定义运行 SQL Server 的服务器组。
- 将个别服务器注册到组中。
- 为每个已注册的服务器配置所有 SQL Server 选项。
- 在每个已注册的服务器中创建并管理所有 SQL Server 数据库、对象、登录、用户和权限。
- 在每个已注册的服务器上定义并执行所有 SQL Server 管理任务。
- 通过唤醒调用 SQL 查询分析器，交互地设计并测试 SQL 语句、批处理和脚本。
- 唤醒调用为 SQL Server 定义的各种向导。

10. SQL 事件探查器

SQL 事件探查器是图形工具，使系统管理员得以监视 Microsoft SQL Server 实例中的事件。可以捕获有关每个事件的数据并将其保存到文件或 SQL Server 表中供以后分析。例如，可以对生产环境进行监视，了解执行速度太慢而妨碍性能的存储过程。使用 SQL 事件探查器可以完成以下操作：

- 监视 SQL Server 实例的性能。
- 调试 Transact-SQL 语句和存储过程。
- 识别执行慢的查询。
- 在工程开发阶段，通过单步执行语句测试 SQL 语句和存储过程，以确认代码按预期目标运行。
- 通过捕获生产系统中的事件并在测试系统中重播它们来解决 SQL Server 中的问题。这对测试和调试很有用，并使得用户可以不受干扰地继续使用生产系统。
- 审核和复查在 SQL Server 实例中发生的活动。这使得安全管理员得以复查任何审核事件，包括登录尝试的成功与失败，以及访问语句和对象权限的成功与失败。

11. 在 IIS 中配置 SQL XML 支持

Microsoft SQL Server 2000 引入了支持 XML 功能的新特性。这些功能组合在一起使 SQL Server 2000 成为可支持 XML 的数据库服务器。这些特性包括：

- 能够使用 HTTP 访问 SQL Server。
- 支持 XDR(XML 数据简化)架构并且能够指定对这些架构的 XPath 查询。
- 能够检索并写入 XML 数据。
- 使用 SELECT 语句和 FOR XML 子句检索 XML 数据。
- 使用 OPENXML 行集提供程序写入 XML 数据。
- 使用 XPath 查询语言检索 XML 数据。
- 增强了 Microsoft SQL Server 2000 OLE DB 提供程序 (SQLOLEDB)，使得可以将 XML 文档设置为命令文本并以流的形式返回结果集。

习题、上机练习与实训 2

一、习题

1. 请为只有一台计算机的用户、有两台计算机的用户或两台以上计算机的用户策划一个 Web 应用开发与运行环境。
2. 安装 JSP 运行环境需要准备哪些软件?
3. JDK 软件的作用是什么? 可以从哪里得到最新版的 Java 平台软件?
4. JDK 安装完成后为什么要配置系统的环境变量? 如何配置?
5. Tomcat 服务器软件的默认发布目录是什么?

二、上机练习

安装并配置 Windows 操作系统下的 JSP 运行环境：

（1）安装 JDK，配置系统的环境变量，测试 JDK 安装是否成功。

（2）安装并配置 Tomcat，安装完成后发布 Tomcat 的默认主页，完成 Tomcat 的启动和停止操作。

（3）创建一个虚拟发布目录，将例题 ex2-01.jsp 存入虚拟目录发布。

（4）安装并配置 SQL Server 2000 数据库系统，并完成停止和启动操作。

三、实训课题

1. 为一个 Web 站点的开发小组规划一个 Windows 200X Server 下的 JSP 工作环境。该组有 10 个成员，客户端可以使用不同的操作系统。请画出该环境的示意图，标明所需的硬件、软件，并规划环境的 IP 地址。

2. 在 Windows 环境下完成 JDK 和 JSP Web 服务器的安装、配置与信息发布。在实训课题 1 的基础上完成以下工作：

（1）网络的硬件连接。

（2）Windows 200X Server 的安装与测试。

（3）客户端操作系统和浏览器安装与测试。

（4）JDK 的安装、配置与测试。

（5）JSP Web 服务器的安装、配置与测试。

（6）TCP/IP 协议的安装与配置。

（7）主页的发布。

第3章 基于Web方式的信息系统开发案例——网上书店

本章结合网上书店的建设,使读者了解一个 Web 站点建设的全貌,这将有助于读者站在全局的高度了解 Web 技术的应用。

使用 Web 方式进行应用系统开发已经成为一种趋势。由于系统采用 Web 方式开发,当进行功能修改、功能扩充等维护性操作时,客户端无须任何改动,使系统具有很高的灵活性和易用性。Web 站点的开发过程应遵循信息系统的开发方法,需要经过系统分析、系统设计、数据库设计、系统详细设计及系统开发、系统维护等环节。

如果开发的 Web 应用系统需要与 Internet 连接,它将是一个面向全球用户的系统,具有用户数量的不确定性。一般来说在用户数量的估算上应有一定的余量,这就要求站点与国际互联网有足够的通信线路带宽。在设计中还需要考虑 Web 站点中的基础 Internet 服务系统(如 DNS 域名解析系统、E-mail 电子邮件系统、FTP 文件服务系统等)的建设。

本案例以网上书店应用为基础模型,涵盖了网上书店的主要业务。为突出主要技术,对系统中的某些细节进行了删节。读者可以在本案例的基础上,开发出符合自己需求的 Web 站点。

"网上书店"的案例将贯穿于本书后续各章节,本章主要介绍系统设计、系统功能设计、数据库设计及部分功能的详细设计。在介绍功能的同时介绍功能实现所使用的技术,以使读者在后续内容学习时了解所学内容在系统中的位置。通过本章的学习,读者将了解应用 Web 方式进行应用系统开发的全过程,了解 Web 站点建设所使用的主流技术。

3.1 系统功能与系统环境

3.1.1 系统功能和使用模式

网上书店的主要业务是开展网上购书活动。系统功能由两个部分组成:前台业务和后台管理。系统通过 Web 网站的形式发布图书消息,客户通过网络访问该网站,查找所需要的图书,办理购书业务。

系统采用 B/S(浏览器/服务器)模式开发,是一个面向互联网用户的虚拟书店,书店

的全部业务在服务器上完成。基本构架如图 3-1 所示。

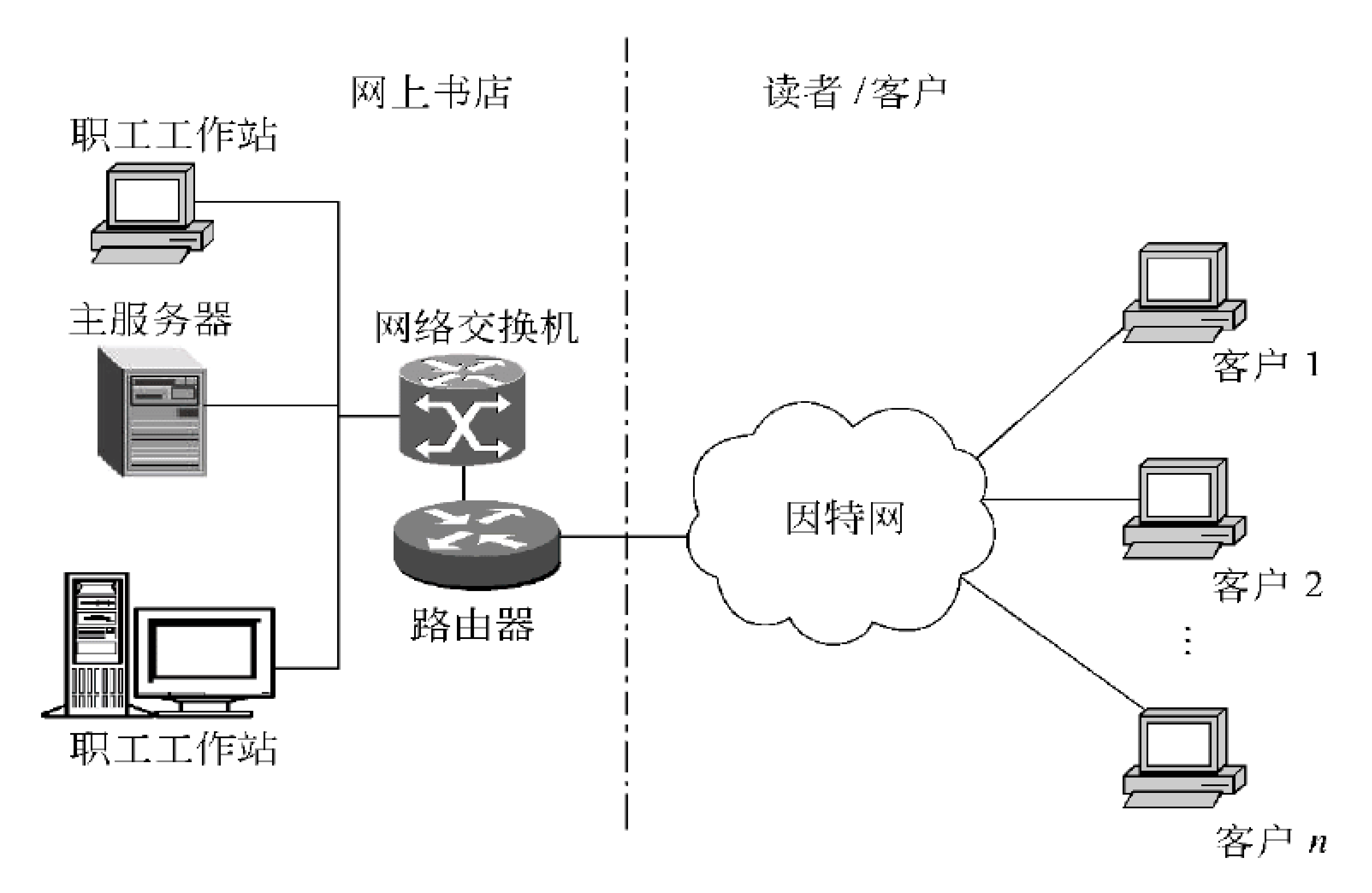


图 3-1　网上书店的 B/S 模式

网上书店的内部网络系统如图 3-1 左部所示，系统使用交换机等网络设备构成书店内部网络系统。系统由主服务器和职工工作站构成。本店职工使用工作站访问主服务器，完成自己的工作任务，其模式也是使用 B/S 模式。网上书店的网络系统通过路由器与因特网连接，读者通过因特网访问书店的主服务器。

3.1.2　系统环境建设

要建设一个可以在互联网上运行的服务系统，必须建立相应的可以和因特网连接的网络环境，步骤如下。

1. 建设内部网络系统

对于规模不大的网络书店，网络系统也不会太复杂。使用交换机将主服务器、职工工作用的 PC 机连接起来就可以工作。

2. 将网络系统与因特网互联

为了提供互联网服务，网上书店必须建立与因特网的永久连接。为此，需要寻找一个 Internet 服务供应商（ISP）。不同的 Internet 服务供应商，提供的服务和价格不尽相同。主要考虑指标是连接速度（带宽），使客户从互联网的任何地点都能较快地访问到它。

3. 确定 IP 地址

要在互联网上提供服务，必须使用互联网上可以访问到的网络地址。互联网上的 IP 地址是由互联网统一分配的，必须由 Internet 服务供应商提供。在网上书店中，至少有一个服务器（图 3-1 中的主服务器）使用合法的静态 IP 地址。在本案例中，动态 IP 地址是

不可用的，当一般用户通过 ADSL 联入互联网时，用户往往只得到一个动态的 IP 地址，这种地址是不固定的，当再次访问互联网时这个地址会改变，因此不能作为永久服务地址使用。

4. 确定域名

客户访问互联网中的网站时，通常不使用 IP 地址，而是使用域名。因此必须建立域名和 IP 地址的对应关系。这种对应关系由域名解析服务器(DNS)完成。域名由统一的组织进行管理。国际上管理域名的组织叫 NIC(网址为 www.internic.net)，读者可以访问这个网站得到进一步的信息。中国管理域名的组织是 CNNIC(网址为 www.cnnic.net.cn)。中国教育和科研计算机网网络信息中心 CERNIC 负责中国教育部门的域名注册(网址为 www.cernic.net.cn)。

5. 完成网络书店与 ISP 的物理连接

ISP 选定后，要解决与选定的 ISP 间的物理连接，即选择通信介质问题。根据需要的不同，有多种可供选择的介质。通信线路的选择应当满足通信双方的要求，因此在选择通信线路时应与 ISP 协商。常用的线路有以下几种：

(1) 光纤线路：光纤线路具有很高的带宽和线路稳定性，但架设困难。光纤线路一般可以提供高达 1000MHz 的线路带宽，可到电信部门租用裸光纤线路使用。现在一般较大的楼宇中都预先铺设和因特网连接的光纤线路，并提供入住厂商的因特网接入服务。

(2) DDN 线路：DDN 是 Digital Data Network(数据传输网)的英文简称，是利用光纤、数字微波、卫星等数字信道，以传输数据信号为主的数字通信网络，可以提供 2MHz 的全透明的数据专线，并承载语音、传真、视频等多种业务。DDN 线路实际上是专用线路，与帧中继相比，使用成本较高。

(3) 帧中继是一种快速分组交换技术，它以面向协议数据的通信规程为基础，依靠智能化的用户终端和高稳定性的传输线路，在用户和网络接口之间提供用户信息的双向透明传输，并提供对多种网络协议的支持。由于帧中继使用分组交换技术，电信线路并不是由某个用户专用，对用户而言线路成本较低。帧中继线路可提供高达 2MHz 的线路带宽。

(4) 卫星信道：卫星信道是一种高质量的数据传输信道。可以提供较宽的带宽和可靠的通信质量。卫星信道可以租用，在需要的情况下，也可以考虑租用卫星信道解决通信问题。为解决双向通信的问题，需要建设卫星接收小站(VSAT)，但初始造价较高。

(5) 无线通信：在有线线路不方便的地方可以考虑使用无线通信的手段完成线路连接。最常用的有无线网桥和无线 Modem。点对点型无线网桥可用来连接两个分别位于不同地点的网络，根据无线设备发射功率的不同，可靠的传输距离各异，一般可支持 10km 左右的距离，如果需要更远距离的传输，可以增加射频放大器，使通信距离达到数十千米。

6. 使用主机托管形式解决与 Internet 的连接问题

互联网数据中心(Internet Data Center，IDC)近几年被全球 ISP 业界认为是第二代

ISP 业务，它除了能提供 Internet 接入服务之外，还能提供电信级的网络资源以及全面的网络管理和应用服务。网上书店的网络环境也可以选择使用主机托管等方式解决。现在市场上有很多 IDC 公司，在 IDC 那里托管自己的主机，解决互联网的连接问题。

3.2 系统设计

系统的开发过程应遵循软件系统的生命周期原则。系统的生命周期即系统从开始开发到系统结束使用的全过程。系统的生命周期包括系统分析、系统设计、数据库设计、系统详细设计、系统开发和代码开发、系统运行及维护等几个重要阶段。要建设一个实际使用的应用系统，首先应当进行必要的系统调查，尽量做到对现有系统具有全面的了解，找出影响系统性能的主要问题，即所谓的系统分析。在充分调查研究的基础上进行系统的设计，并充分征求系统应用人员的意见。系统的分析和设计可能需要多次反复讨论研究，力求做到符合应用系统各方面人员对系统的使用需求。系统设计应产生系统设计说明书，交应用系统的使用者确认。系统设计说明书是进行系统建设和实施的规定性文件。经应用系统的使用者确认后，原则上不再进行修改。如有重大修改则应重新进行系统设计，少量修改可以在系统开发完成后在系统维护中进行。系统设计说明书完成后进入程序代码开发和调试阶段。开发调试完成后交最终用户验收和试运行。试运行合格后进入系统的维护阶段直至系统生存周期的结束，旧的系统被新的系统取代，上升到一个新系统的生命周期。

“网上书店管理信息系统”的设计也将遵循这些原则。由于本书的主要目的是讲解 Web 技术的应用，所以在本章给出设计结果，在以后的各章中按照本章的设计结果，以实例的形式给出部分功能的实现代码，读者既可以从实例中学习 Web 技术的应用，也可以了解该技术在整个系统开发中的地位。

3.2.1 系统设计原则

1. 实用

系统设计的重要原则是系统的实用性，系统必须符合用户的需求。应注重采用先进、成熟而实用的技术，使系统建设的投入产出比最高，产生良好的社会效益和经济效益。

2. 可靠

系统中的软硬件及信息资源应满足可靠性设计要求，保证系统长期安全地运行。

3. 先进

在实用的前提下，尽可能跟踪国内外先进的计算机软硬件技术、信息技术和网络通信技术，使系统具有较高的性能指标。

4. 可扩充

系统的软硬件具有升级扩充余地，不因系统的扩充、升级或改型使原有系统失去作用。

5. 安全

系统应具有必要的安全保护和保密措施，有很强的应对计算机犯罪和防范病毒的能力。

6. 用户界面友好

贯彻面向最终用户的原则，设计并制作友好的用户界面，使用户的操作简单直观，易于学习掌握。

7. 健壮

系统具有较强的抗干扰能力和容错能力。对各类用户的误操作或异常情况应有提示或自动消除能力。

8. 可自适应

系统应对不断发展和完善的业务需求和开发方法具有一定的适应能力。

3.2.2 系统需求分析

"网上书店管理信息系统"是网上书店使用的管理信息系统，读者通过网络连接访问网上书店，查找读者需要的图书，找到后放入购物车并填写读者相应的信息。书店管理人员定期查看购物车的情况，按照读者的要求将读者申购的图书送达读者，完成购书活动。

随着因特网的迅速发展，网民数量的急剧增长，形成巨大的消费市场。图书作为一种特殊商品，特别适合网络销售活动，目前已具规模的如当当网、亚马逊、中国图书网等。网上书店以其营业成本低、服务时间长、方便用户等特点成为一种重要的图书销售渠道。作为一种开发实例，网络书店在电子商务中具有一定的典型意义。本书的教学将围绕着网上书店的开发展开。通过该实例的展开，讲解 Web 应用系统中的各种技术问题。

该系统的使用人员主要有两种：网络购书者和书店工作人员。

3.2.3 网络及服务器的选择

网上书店是一个网络上的虚拟书店，由内部网络和外部网络两部分组成。网上书店内部网络系统是一个局域网，由交换机等网络设备组成，满足书店职工工作的需要。内部网络系统通过远程线路连接到互联网，保证读者对网站的访问。本店职工通过内部网络使用该系统，读者通过因特网访问本店的主服务器（网上书店 Web 站点）。为保障系统的

安全，使用防火墙隔离内部网络和外部服务，其网络连接如图 3-2 所示。

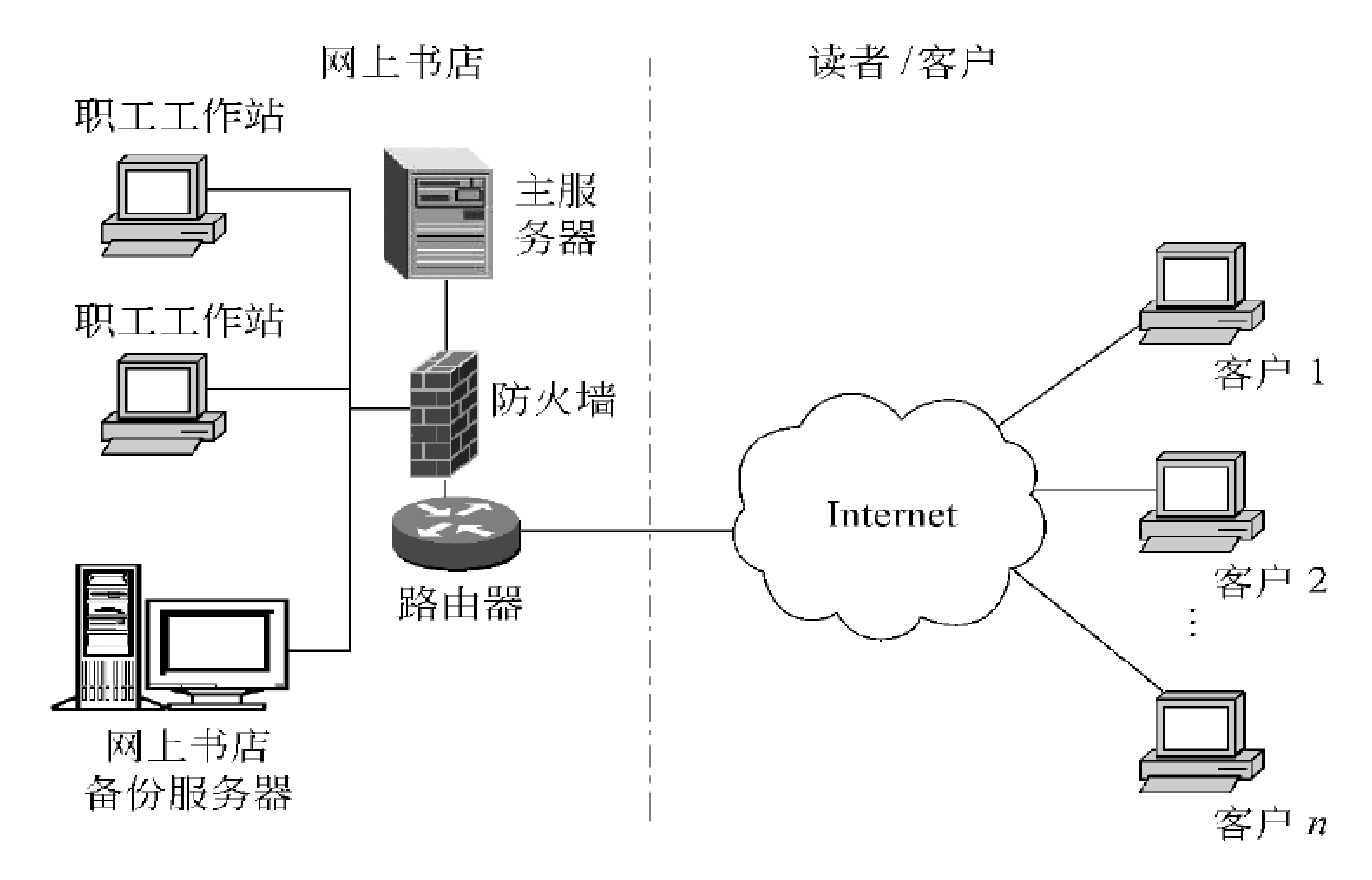


图 3-2　网上书店网络连接图

考虑到图书信息的电子化和多媒体化的趋势，网上书店内部网络带宽采用 100MHz 或 1000MHz 以太网，保证系统的可扩充性。

网上书店管理信息系统网站设置主服务器和备份服务器，当主服务器受到攻击出现故障时，备份服务器中的数据可以保证系统的迅速恢复。服务器应采用性能价格比较高的服务器。

3.2.4　系统软件结构

网上书店管理信息系统采用先进的 Browser/Server/Database Server 模式，服务器端采用 Web 方式进行应用系统开发，客户通过浏览器访问“网上书店管理信息系统”，服务器端使用应用逻辑服务和数据库服务两层，与客户端形成三级系统结构。采用这样的三层结构，具有结构清晰、便于维护、运行速度高等特点。

3.3　系统功能设计

3.3.1　“网上书店管理信息系统”的功能

“网上书店管理信息系统”的主要功能包括客户端处理和管理端处理。客户端处理提供客户进入书店后的各种服务，包括图书展示、用户身份验证、购书车三个功能模块。系统提供方便的图书查找工具，找到所需图书后，进行网上订购，将图书放入购物车，确认用户身份，直至用户决定购买(下订单)。

管理端功能解决书店内部的处理问题，包括图书管理、读者管理、订单管理和本店职工的工作职责与权限管理等。网上购书系统功能结构如图 3-3 所示。

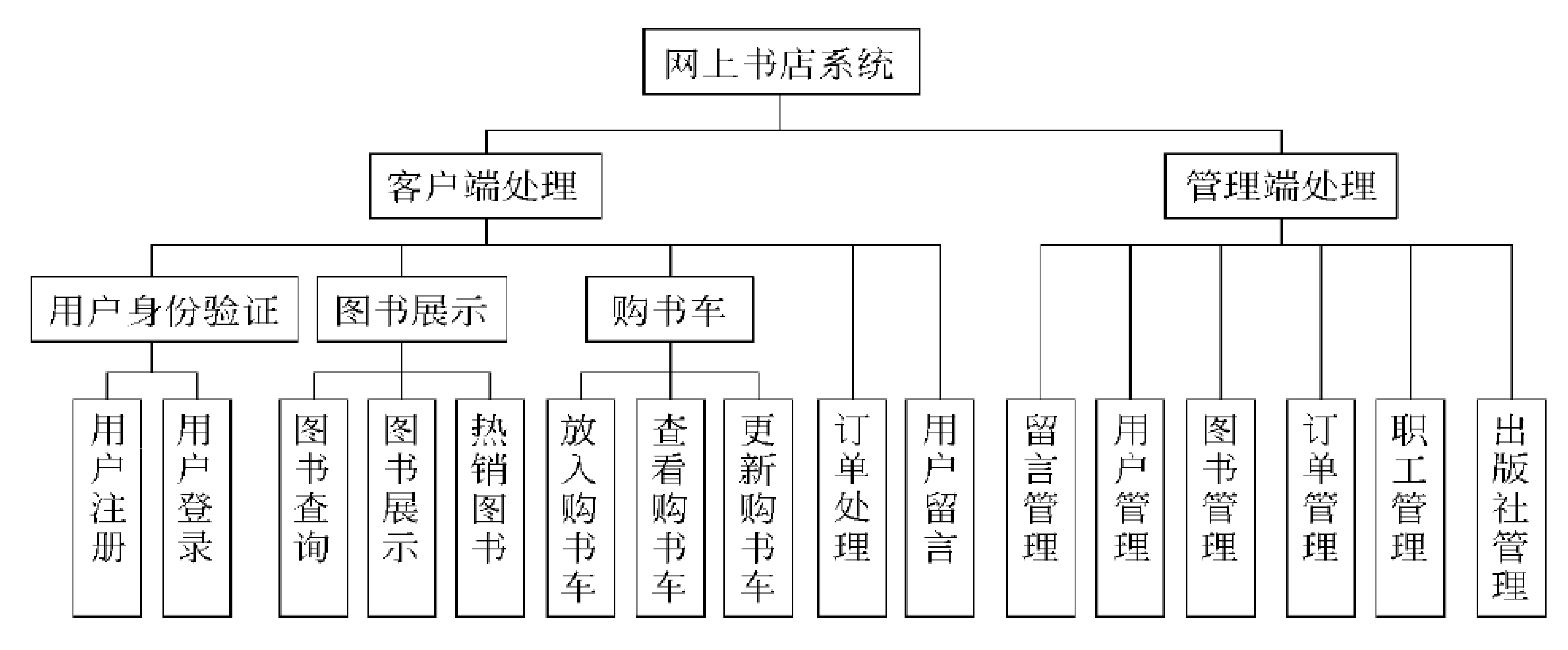


图 3-3　网上书店系统功能结构图

各模块功能如下所述。

1．客户端处理模块

1）图书展示功能

图书展示功能提供各种方便的浏览图书工具，包括新书展示和图书查询。

新书展示：从数据库中取出标记为新书的图书，放到页面上。

图书查询：提供方便的图书检索工具。通过输入所要查询图书的书名、作者、出版社、关键字等信息查询图书，读者还可以进行模糊查询。

2）购书车

购书车功能包括放入购书车、显示购书车、清空购书车等功能。

放入购书车：读者看到需要的图书后，单击图书条目旁的购书车图标，将该书放入购书车。

显示购书车：单击购书车图标随时查看购书车中的情况。

清空购书车：查看购书车中的情况并可以清空该购书车。

继续购书：查看购书车后返回主菜单继续购书。

去收银台：购书完成后下订单。

3）订单处理

把图书放入购书车并决定购买时，单击订单按钮进入订单处理模块，系统只对注册了的用户提供图书订购服务。如果是合法用户，显示订单表格，要求用户填写相应的订单信息。系统检查订单信息的完整性和正确性，如果正确，将此订单存入数据库。

4）用户身份验证

用户身份验证包括用户注册和用户登录两个模块。

用户注册：如果用户尚未在系统中注册，则进入用户注册窗口，输入注册信息。要求输入用户的账号、密码及个人信息。系统将检查这些信息，不合法时要求重新输入。合法时将用户信息存入数据库。

用户登录：只有合法注册了的用户才可以订购图书。进入订单处理后，系统要求输入用户的账号和密码，进行登录并检查信息。用户登录后可以修改用户的账号和密码，查

看订单的处理情况及订单历史记录。也可查看留言的处理情况。

5）读者留言板

登录的客户可以留言。通过留言板，把需要的图书、要求和建议等记录下来，与图书馆管理人员交流。

2．管理端处理模块

管理人员进入系统时要进行身份检查，正确时方可进入。

1）用户管理

读者注册后用户的状态代码为未激活状态。管理人员在处理订单时应对用户身份的正确性进行验证（电话、短信等确认），如果正确，将用户状态改为激活，打印客户清单。

2）职工管理

职工管理主要处理职工的账号和密码。职工可以修改自己的账号和密码。

3）订单管理

职工随时检查订单情况，查找未处理订单。找到未处理订单后，检查用户状态，如果是激活状态，则查找图书架位，准备配送。如果用户状态未激活，则进入用户管理模块进行激活处理。图书配送后，图书数量改变，并将订单状态改为完成。

4）图书管理

书店进书后，管理员将图书信息和图书所在架位情况填入数据库中。

5）用户留言管理

处理用户留言信息，并将处理日期、处理人和处理意见填写到留言表中。

3.3.2 业务流程设计

网上书店管理信息系统的业务总流程如图3-4所示。图中包含了网上书店客户端和服务器端的业务流程关系。本书中将逐步讲解这个系统的主要处理逻辑。

3.3.3 用户界面设计

用户界面是用户使用系统的主要工具，必须精心设计，使系统达到最佳的使用效果。系统界面主要有两个部分组成：客户端页面和管理端页面。

1．客户端主页面

当用户登录网上书店时，进入系统的主页面。系统主页面如图3-5所示。

主页面中有如下功能区。

公告栏：书店发布的公告信息。

精品图书栏：展示当前热销的精品图书。

最新图书栏：展示最新图书。

图书查找栏：提供用户快速查找图书的功能。读者可以通过书名、作者、出版社等信

订单库
订单细节库
新用户注册
读者表
我的订单查询
订单显示
留言库
用户登录
通过
留言反馈查询
留言反馈意见
更多精品新书
新书精品图书信息
详细信息
新书精品详细信息
留言处理
用户
书库
图书管理
职工管理
职工库
网上书店主页
用户功能选择处理
图书查找
图书列表
管理员
出版社管理
服务员
高级图书查找
读者表
订单管理
出版社库
订单库
购物车
购物车列表
订单
订单细节库

图 3-4　系统业务流程总图



图 3-5　客户端主页面

息进行图书快速查找。

用户登录、注册栏：用户购书时，系统检查用户的登录情况，只有登录的用户才可以

购买图书，用户应先完成用户注册，填写必要的个人信息，得到用户名和密码。注册后书店将长期保留用户的注册信息。用户登录系统时输入自己的用户名和密码。登录用户可以查看自己的订单情况，也可以浏览自己的读者留言的反馈信息。

购书车：客户可在任何时间将图书放入购书车，进入购书车查看并修改购书信息，下订单（去收银台），完成购书活动。书店员工将根据读者订单配送图书。

2. 管理端主界面

系统管理端软件由本店职工使用，完成书店的各种管理业务。为保证系统的安全，管理员进入管理系统时必须输入相应的口令。输入界面如图 3-6 所示。



图 3-6　管理员登录界面

职工输入了正确的用户名和口令后进入管理员主界面。系统管理端主界面如图 3-7 所示。

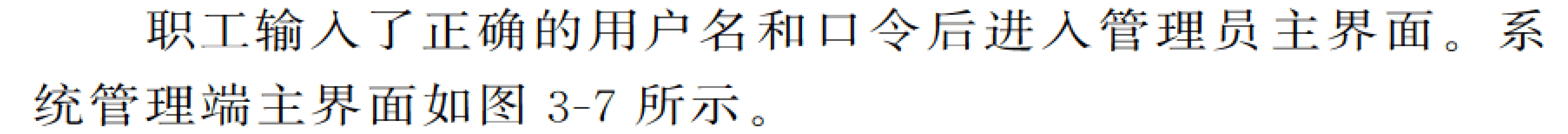




图 3-7　网上书店管理员主界面

3. 页面设计所使用的技术

界面的布局和页面制作：可以使用 Dreamweaver、Flash 和 FrontPage 等软件，但是不管使用什么软件，Web 编程的基础都是 HTML，这部分内容将在第 4 章介绍。

数据库信息发布与存储技术：用户登录时在界面提交的用户名和密码必须与数据库 userinfo 表中的信息一致，才允许用户进入系统的主界面，这需要查询数据库信息。读者在查询图书库时需要查询 book 表中的信息，找到所需图书后，系统将把该书的信息返回给读者。读者订购图书时系统将读者订购图书的信息写入 orderform 订单主表和 orderdetail 订单子表。本书将在第 7、8、9、10 章重点介绍应用较为广泛的 JSP 技术。

其他界面，如图书列表界面、购书车界面和客户留言等，将在后面详细介绍。

3.4 数据库设计

网上书店使用 SQL 数据库，数据库名称是：bookshop。

数据库是应用系统的核心，应当精心设计，以保证系统的正确、可靠、高效运行。本案例的数据库共使用 7 个表格。

1. 图书表(见表 3-1)

表 3-1 图书表(表名:book)

字段	类型	中文名	主键/外键	备注
bookid	Varchar(50)	书号	pk	
bookname	Varchar(200)	书名		
author	Varchar(50)	作者		
publisherid	Int	出版社 id		
pubdate	datetime	出版日期		
category	Varchar(50)	分类码		
price	float	书价		
content	text	内容简介		
type	Int	新书，精品标志		0：一般 1：新书 2：精品
quantity	Int	库存数量		
place	Varchar(50)	图书所放的位置		
picture	Varchar(50)	图片文件名		

2. 用户表(见表 3-2)

表 3-2 用户表(表名:userinfo)

字段	类型	中文名	主键/外键	备注
userid	Varchar(50)	用户名	pk	
username	Varchar(50)	姓名		
password	Varchar(50)	密码		
gender	Varchar(10)	性别		
address	Varchar(200)	住址		

续表

字　段	类　型	中　文　名	主键/外键	备　注
email	Varchar(50)	E-mail		
phone	Varchar(50)	联系电话		
postcode	Varchar(50)	邮编		
state	Int	用户状态		0：未激活 1：激活 2：重要

3. 订单主表(见表 3-3)

表 3-3　订单主表(表名:orderform)

字　段	类　型	中　文　名	主键/外键	备　注
orderid	Varchar(50)	订单号	pk	
orderdate	datetime	订单日期		
userid	Varchar(50)	用户 id		
totalnum	Int	所购图书数量		
totalamount	float	总金额		
payment	Varchar(50)	付费方式		
deliver	Varchar(50)	送货方式		
receiver	Varchar(50)	收货人		
address	Varchar(200)	收货地址		
phone	Varchar(50)	联系电话		
postcode	Varchar(50)	收件人邮编		
state	Int	订单状态		0：未处理 1：发货 2：完成

4. 订单子表(见表 3-4)

表 3-4　订单子表(表名:orderdetail)

字　段	类　型	中　文　名	主键/外键	备　注
id	Int	序号	pk	自动增长
orderid	Varchar(50)	订单号		
bookid	Varchar(50)	书号		
bookname	Varchar(200)	书名		

续表

字　　段	类　　型	中　文　名	主键/外键	备　　注
publisher	Varchar(200)	出版社		
unitprice	float	单价		
ordernum	Int	订购数量		

5. 用户留言(见表 3-5)

表 3-5　用户留言表(表名:notes)

字　　段	类　　型	中　文　名	主键/外键	备　　注
id	Int	序号	pk	自动增长
userid	Varchar(50)	用户 id		
subject	Varchar(200)	留言主题		
date1	datetime	留言日期		
context	Text	留言内容		
employeeid	Varchar(50)	处理人 id		
date2	datetime	处理日期		
advice	Text	处理建议		

6. 职工表(见表 3-6)

表 3-6　职工表(表名:employee)

字　　段	类　　型	中　文　名	主键/外键	备　　注
employeeid	Varchar(50)	职工账号	pk	
name	Varchar(50)	职工姓名		
password	Varchar(50)	密码		
gender	Varchar(10)	性别		
address	Varchar(200)	住址		
email	Varchar(50)	E-mail		
phone	Varchar(50)	联系电话		
task	Varchar(50)	工作任务		

7. 出版社表(见表 3-7)

表 3-7　出版社表(表名:publisher)

字　段	类　型	中　文　名	主键/外键	备　注
publisherid	Int	出版社编号	pk	自动增长
name	Varchar(50)	出版社名称		
linkman	Varchar(50)	联系人		
address	Varchar(200)	出版社地址		
email	Varchar(50)	E-mail		
phone	Varchar(50)	联系电话		
website	Varchar(200)	网址		

3.5　代码设计与实现

网上书店管理信息系统应用 JSP 技术实现,读者在学习本书的内容时可以参考这些程序。

代码功能见表 3-8 和表 3-9。

表 3-8　客户端程序清单及功能表

功　能	程序名称	功能描述
主页	index. jsp	显示书店主页
精品图书	excellent. jsp	显示精品图书
新书架	newbook. jsp	显示新书架图书
书目查找	booksearch. jsp	书目查找
书目查找清单	booklist. jsp	书目查找清单
我的订单	myorder. jsp	订单查看
购书车	shoppingcart. jsp	显示购书车
购书车	addtocart. jsp	把书放入购书车
购书车	increaseCart. jsp	增加所购图书数量
购书车	clearcart. jsp	清空购书车
购书车	decreaseCart. jsp	减少所购图书数量
购书车	delfromcart. jsp	取消某书目
读者留言	leaveword. jsp	显示留言板
读者留言	leaveword2. jsp	把留言写入数据库

表 3-9　管理端程序清单及功能表

功　　能	程 序 名 称	功 能 描 述
图书管理	booklist. jsp	显示图书清单
图书管理	bookadd. jsp	添加图书
图书管理	bookedit. jsp	编辑图书信息
图书管理	bookdelete. jsp	删除图书
图书管理	booksearch. jsp	查找图书
用户管理	userinfolist. jsp	显示用户清单
用户管理	userinfoedit. jsp	编辑用户信息
用户管理	userinfoadd. jsp	添加用户信息
用户管理	userinfodelete. jsp	删除用户
用户管理	userinfosearch. jsp	查找用户
订单管理	orderlist. jsp	显示订单清单
订单管理	orderedit. jsp	编辑订单信息
订单管理	ordersearch. jsp	查找订单
留言管理	noteslist. jsp	显示留言清单
留言管理	notesedit. jsp	编辑留言信息
留言管理	notesdelete. jsp	删除留言
留言管理	notessearch. jsp	查找留言
职工管理	employeelist. jsp	显示职工清单
职工管理	employeeadd. jsp	添加职工
职工管理	employeeedit. jsp	编辑职工信息
职工管理	employeedelete. jsp	删除职工
职工管理	employeesearch. jsp	查找职工
出版社管理	publisherlist. jsp	显示出版社清单
出版社管理	publisheradd. jsp	添加出版社
出版社管理	publisheredit. jsp	编辑出版社信息
出版社管理	publisherdelete. jsp	删除出版社
出版社管理	publishersearch. jsp	查找出版社
显示版权信息	Bottom. jsp	显示版权信息
显示出错信息	error. jsp	显示出错信息
购书车的数据结构	CartBean. java	购书车的数据结构

3.6 网上书店的安装及使用

为使读者在学习过程中易于理解，并参考使用，将系统适当剪裁并移植到计算机环境中，读者可以将网上 bookshop 中的内容复制到 Tomcat 的发布目录下即可运行。

安装操作步骤如下：

(1) 安装 JDK。

(2) 安装 Tomcat。

(3) 安装 MS SQL Server 2000 个人版。

(4) 将 bookshop 下的数据库备份文件导入到数据库中，数据库取名为 bookshop。

(5) 创建数据源，在"控制面板→管理工具→数据源"中建立与该数据库对应的数据源，数据源取名为 bookshoplk。

(6) 将 bookshop 整个文件夹复制到 tomcat 发布目录 webapps 下。

(7) 以上步骤完成后，重启 tomcat，在浏览器地址栏目中输入：

http://localhost:8080/bookshop 客户端(初始账号和密码:pds/pass)

http://localhost:8080/bookshop/admin 管理端(初始账号和密码:admin/pass)

习题与实训 3

一、习题

1. 一个基于 Web 的应用系统开发主要过程有哪些？
2. 系统设计的主要原则是什么？
3. 系统环境建设的主要步骤有哪些？如何选择通信线路？
4. 完成用户身份验证子系统的分析与设计。
5. 完成图书展示子系统的分析与设计。
6. 完成用户留言子系统的分析与设计。
7. 完成购书车子系统的分析与设计。
8. 完成职工管理子系统的分析与设计。

二、实训课题

完成一个网上购物(例如计算机、笔记本电脑、光盘、旅游用品等)管理信息系统的分析与设计。

第 2 篇　Web 客户端程序设计基础

Web 技术中很重要的一个模块是 Web 编程技术。Web 程序设计与一般意义上的程序设计有所不同,专业人员与非专业人员都可以进行。它的开发工具简单好用,语法结构简洁易掌握,不需要太多的专业知识,特别适合非专业人员使用,这也是 Web 技术易于普及并受大众欢迎的重要原因之一。本篇主要介绍 Web 客户端程序设计技术,它也是 Web 服务器端程序设计的基础。通过第 2 篇的学习,读者将掌握 Web 客户端编程技术,完成 Web 客户端的应用开发工作。第 2 篇主要内容包括:

第 4 章　HTML
第 5 章　CSS
第 6 章　JavaScript

对于已经掌握程序设计方法的技术人员,可以快速通过第 2 篇,掌握 Web 编程的要点,进入后续内容的学习。

第4章 HTML

Web方式的数据库信息发布是基于HTML的，一般将执行与数据库交换信息的ASP/ASP.NET、PHP或JSP语句嵌入HTML文档，由服务器端的引擎执行，完成数据库的信息发布工作。网页是以HTML格式写成的，HTML通过标记(Tag)式指令，将影像、声音、图片和文字等连接并显示出来。HTML是符合SGML(Standard Generalized Markup Language，标准通用标记语言)语法的一种固定格式的超文本标记语言，当浏览HTML页面时，浏览器将自动解释标记的含义，并按标记指明的格式展示内容。

4.1 HTML概述

HTML是一种控制页面内容显示的标记语言，按照HTML语法编写的文件称为HTML文件。可以使用任意的文本编辑器编写HTML文件，以纯文本形式存储，并以htm或html为扩展名。文件编写完成后，可在浏览器中查看效果。

4.1.1 HTML入门——一个简单HTML案例

例4.1 请读者一起来制作具有跳转功能的简单网页。本案例具有两个页面，文件名为：ex4-01.html和ex4-01_1.html。在浏览器的地址栏目输入文件ex4-01.html的URL，浏览器将显示文件ex4-01.html的内容，在该页面单击超链接，页面将跳转到ex4-01_1.html页面，这是网页最基本的功能。请按以下步骤进行。

1. 编写HTML代码

使用简单的文字编辑器，如Notepad(记事本)和WordPad(书写板)等都可以胜任。请在编辑器中输入以下两个HTML文件的代码。

1) 文件名ex4-01.html

```
<html>
<head>
    <title> 简单的 html 案例 </title>
```

```
</head>
<body>
    欢迎！
    请单击<a href="ex4-01_1.html">Hello World！ </a>
</body>
</html>
```

2）文件名 ex4-01_1.html

```
<html>
<head>
      <title>Hello World！ </title>
</head>
<body>
      祝贺你，初试成功！
</body>
</html>
```

以扩展名为 html 保存文件。在只支持三个字母作后缀的操作系统中，它的扩展名是 htm。

2．页面测试

在浏览器中显示页面 ex4-01.html 运行结果，如图 4-1 所示。

在图 4-1 页面中单击“Hello World！”，页面将跳转至代码 ex4-01_1.html 所显示的页面，如图 4-2 所示。



图 4-1　页面代码 ex4-01.html 在浏览器中的显示



图 4-2　页面跳转到 ex4-01_1.html

3．HTML 标记功能简介

以页面代码 ex4-01.html 为例说明 HTML 标记的简单使用。

第 1 行的<html>和最后 1 行的</html>表示 HTML 文档的开始和结束。

第 2 行的<head>和第 4 行的</head>指示 HTML 文档文件头的开始与结束。

第 3 行的<title>简单的 html 案例</title>，在浏览器标题栏显示标题“简单的 html 案例”。

第 5 行的<body>和第 8 行的</body>说明该页面的文件体的开始与结束，文件体中的内容在浏览器中显示。

第 6 行和第 7 行是 HTML 文档正文，是在浏览器中显示的内容。第 7 行中的Hello World! ，标记<a>说明其后的文字是一个超链接，单击该文字，页面将跳转至"ex4-01_1.html"页面。

4.1.2 HTML 文件的结构

HTML 文件总是以标记<html>开始，</html>结束。用标记<head></head>和<body></body>把文档分为两部分。在<head>与</head>之间的是文件头，文件头内包含关于文件的说明信息，它们不和文件一起显示。在<body>和</body>之间的是文件要显示的内容，包括有标题、段落、列表、图形和超文本链接等。HTML 文件结构如图 4-3 所示。

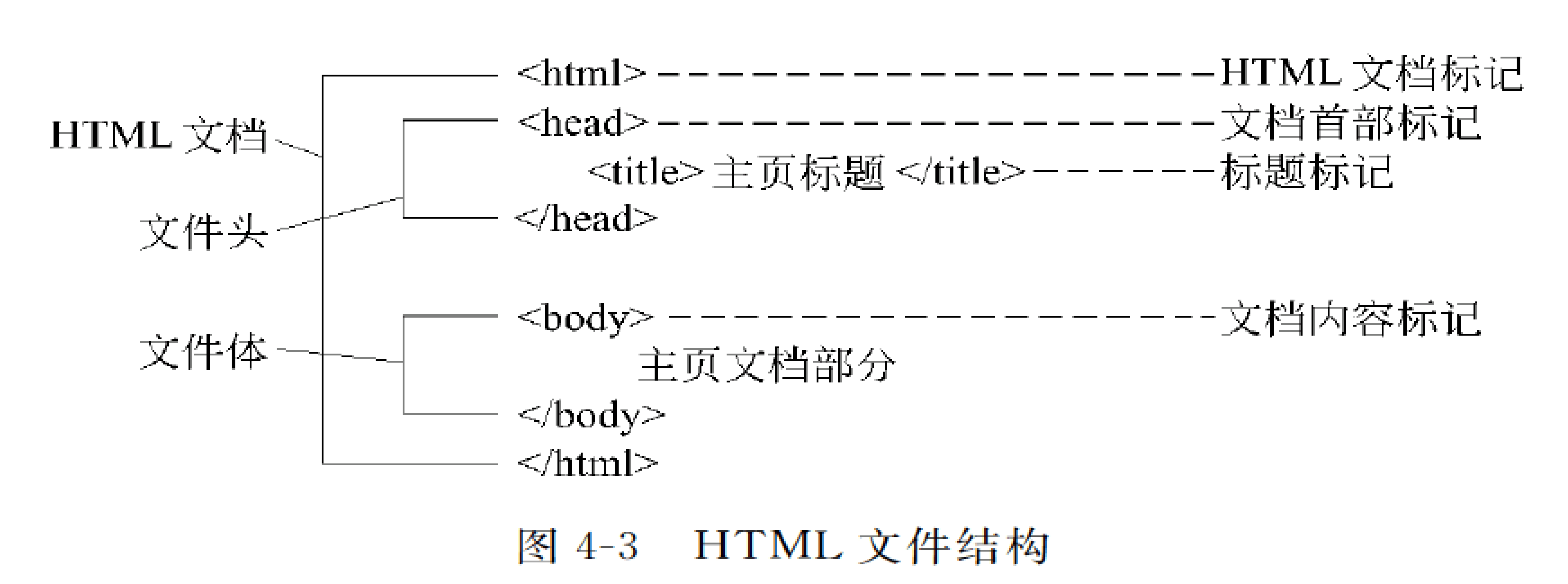


图 4-3　HTML 文件结构

4.1.3 HTML 的标记和元素

HTML 文件的所有控制语句称为标记，标记在一对尖括号之间，格式如下：

<标记名称>HTML 元素</标记名称>

例如，

<h2> 你好！</h2>

- 标记大多成对出现，由始标记<标记名称>和尾标记</标记名称>组成。<标记名称>通知浏览器开始执行该标记名称指示的功能，</标记名称>说明该功能到此结束。也有的标记没有尾标记，如换行标记
既没有尾标记也没有 HTML 元素，当浏览器遇到
标记时换行显示文本。
- 由标记控制在浏览器中显示的内容称为 HTML 元素，例如，<h2> 你好！</h2>中的文字"你好！"。
- 标记与字母的大小写无关，例如<BODY>与<body>的作用是一样的。
- 标记可以联合使用，也可以嵌套使用。
- 标记可以带有一个或多个属性参数。格式如下：

<标记名称 属性 1="属性值 1" …属性 n="属性值 n"> HTML 元素</标记名称>

- 注释标记<！--注释内容-->，注释内容不在浏览器中显示，只供阅读页面代码时帮助理解使用。

4.1.4 HTML 页面结构标记

1．文件标记<html>…</html>

在 HTML 文档的最外层，表示 HTML 文件的开始与结束。

2．文件头标记<head>…</head>

在文件头标记<head>与</head>之间保存说明整个文件的综合信息。它包含如下标记。

1）文件标题标记<title>…</title>

是文件头中的最常用的标记，它给出文件的总标题，在浏览器标题栏中显示。标题应简明扼要，切中文件的主题。格式如下：

```
<title>标题</title>
```

2）<base href="URL">基地址标记

用来指定本网页中超链接的基准路径，又称基锚。使用该标记可以简化页面中超链接的编写，只需要写出相对基准路径的相对路径即可。

3）<meta>相关资料标记

用来说明与本文件有关的资料信息，包括资料名称 name 和内容 content 属性。

4）<link>标记

指明本网页需要其他资源的情况，显示作者信息、相关检索信息等。

5）CSS 样式标记<style>…</style>

文件头标记应用示例如下：

```
<html>
<head>
    <title>简单的 html 案例</title>
    <base href="http://www.buu.edu.cn/">
    <meta name="author" content="july">
</head>
<body>
    欢迎！
    请单击<a href="ex4-01_1.html">Hello World！ </a>
</body>
</html>
```

3．文件主体标记<body>…</body>

在 HTML 中标明文档的主体，是 HTML 文档的主要部分。<body>表示文档的开

始,</body>表示文档的结束。<body>标记可以使用属性使页面带有背景颜色或背景图案,它的主要属性如下。

- bgcolor:设置页面背景颜色。例如,<body bgcolor =blue>,<body bgcolor = #aa88cc>。
- background:背景图案或图像文件的 URL。例如,<body background ="back.gif">。
- text:设置网页文字的颜色。例如,<body text = red>。
- alink、vlink 和 link:用来控制超链接文字的颜色,link 属性用来设置尚未链接过的超链接文字的颜色;vlink 属性用来设置已经链接过的超链接文字的颜色;alink 属性用来设置当鼠标单击超链接后超链接显示的颜色。例如,<body alink="green" vlink="red" link="blue">。

颜色用 6 位十六进制的红-绿-蓝(red-green-blue,RGB)值表示,rrggbb。常用颜色及其对应的十六进制数对照见表 4-1。

表 4-1 常用颜色的十六进制数对照表

颜　色	十六进制数	颜　色	十六进制数
black(黑)	#000000	gray(灰)	#808080
navy(深蓝)	#000080	silver(银灰)	#C0C0C0
blue(蓝)	#0000FF	red(红)	#FF0000
teal(墨绿)	#008080	magenta(洋红)	#FF00FF
green(绿)	#00FF00	yellow(黄)	#FFFF00
cyan(青)	#00FFFF	white(白)	#FFFFFF

4.2 HTML 页面修饰标记

页面修饰标记主要用于控制页面的段落,显示字符的大小、颜色、字体和属性等。

4.2.1 标题文字标记<hn>…</hn>

标题文字标记用来标示页面中的标题文字,被标示的文字将以粗体形式显示。HTML 定义了六级标题文字,*n* 的范围是 1~6,<h1>标记的字体最大最黑,其余的依次小下来,<h6>最小。它们需要与尾标记一起使用。

语法:

```
<hn align=left|center|right> 标题文字 </hn>
```

其中,align 属性用来控制标题文字的对齐方式:left——左对齐(默认对齐方式);center——居中;right——右对齐。

4.2.2 文字样式标记…

文字样式标记利用其属性来控制文字的字体、大小和颜色。

语法：

** 文字 **

<font>标记属性的说明见表 4-2。

表 4-2 <font>标记属性

属　性	功　能	应用举例
face	设置文字的字体。如果指定的文字在用户系统中不存在，则使用默认字体	<font face="楷体">
size	设置字体的大小，分为 7 级，等级 7 最大，默认值是 3	<font size=6>
color	设置字体的颜色	<font color ="red">

4.2.3 特定文字样式标记

HTML 中的一些标记可以使文字以特定的样式显示，这些标记分为两类：物理类型和逻辑类型。物理类型的标记直接指定文本显示的具体样式，例如，显示为粗体、斜体或下划线等。逻辑类型的标记说明文本的用途，进而决定文本的样式，例如突出显示、按地址显示等。常用的描述特定文字样式标记见表 4-3。

表 4-3 字体标记

文本样式	描述标记	类　别
粗体		物理
斜体	<i></i>	物理
下划线	<u></u>	物理
删除划线	<s></s>	物理
上标		物理
下标		物理
大字体	<big></big>	物理
小字体		物理
重点突出显示(粗体)		逻辑
突出显示(斜体)		逻辑
电子邮件和网址(缩小+斜体字)	<address></address>	逻辑
按代码显示(缩小字体)	<code></code>	逻辑

4.2.4 段落标记

HTML 文档中的空格、Tab 符、回车换行符等，在浏览器中不起作用，必须要使用标记，才能使文章分出段落，显出层次。

1. 段落标记<p>

语法：

<p align=left|center|right>段落文字</p>

使用<p>标记实现文章段落与段落之间的分隔，段落的前后都有空行。<p>标记表示新的一段开始。</p>表示段落的结束，常常可以省略。

2. 换行标记

在使用<p>标记分段时，在段落之间有一空行。如果不希望出现空行，可以使用
换行标记。当浏览器遇到
标记时会另起一行，中间不插入空行。

3. 水平线标记<hr>

语法：

<hr 属性=属性值>

浏览器遇到<hr>水平线标记，会在页面上画出一条水平线。水平线可以划分页面显示内容，使页面内容更加清晰醒目。<hr>标记的属性用来控制水平线的样式，常用的属性见表 4-4。

表 4-4　<hr>标记的属性

属　　性	功　　能	示　　例
size	水平线的粗细，以像素为单位，默认值是 1	<hr size=6>
width	水平线的宽度，可以以像素为单位，也可以用对屏幕的百分比表示，默认值为 100%	<hr width =40%> <hr width =80%>
align	水平线对齐方式，可取值为：left、center 或 right，默认值是 center	<hr align =right>
color	水平线的颜色	<hr color ="red"> <hr color =#bb6688>

4. 预格式化标记<pre>

预格式化标记<pre>可以使 HTML 文档中的空格、Tab 符、回车换行符起作用。它与尾标记</pre>一起使用。

4.2.5 页面修饰标记应用案例

例 4.2 代码 ex4-02.html 说明了页面修饰标记的使用。代码清单如下：

```
<html>
<head>
     <title>页面修饰标记应用</title>
</head>
<body bgcolor=#f0f0f0>
<h2 align="center" >显示 2 号标题字</h2>
<center>
<p>
<font color="blue" face="隶书" size=5>显示蓝色隶书字体</font>
<br><s>显示删除划线</s>
<address>aaaa@buu.com.cn(显示地址)</address>
<hr color ="red">
显示上标：x<sup>3</sup><br>
显示下标：x<sub>2</sub>
</center>
<hr color =#00FFFF size=4>
<pre>
                    预格式化标记应用
                          *
                        *   *
                          *
</pre>
</body>
</html>
```

在浏览器中显示效果如图 4-4 所示。

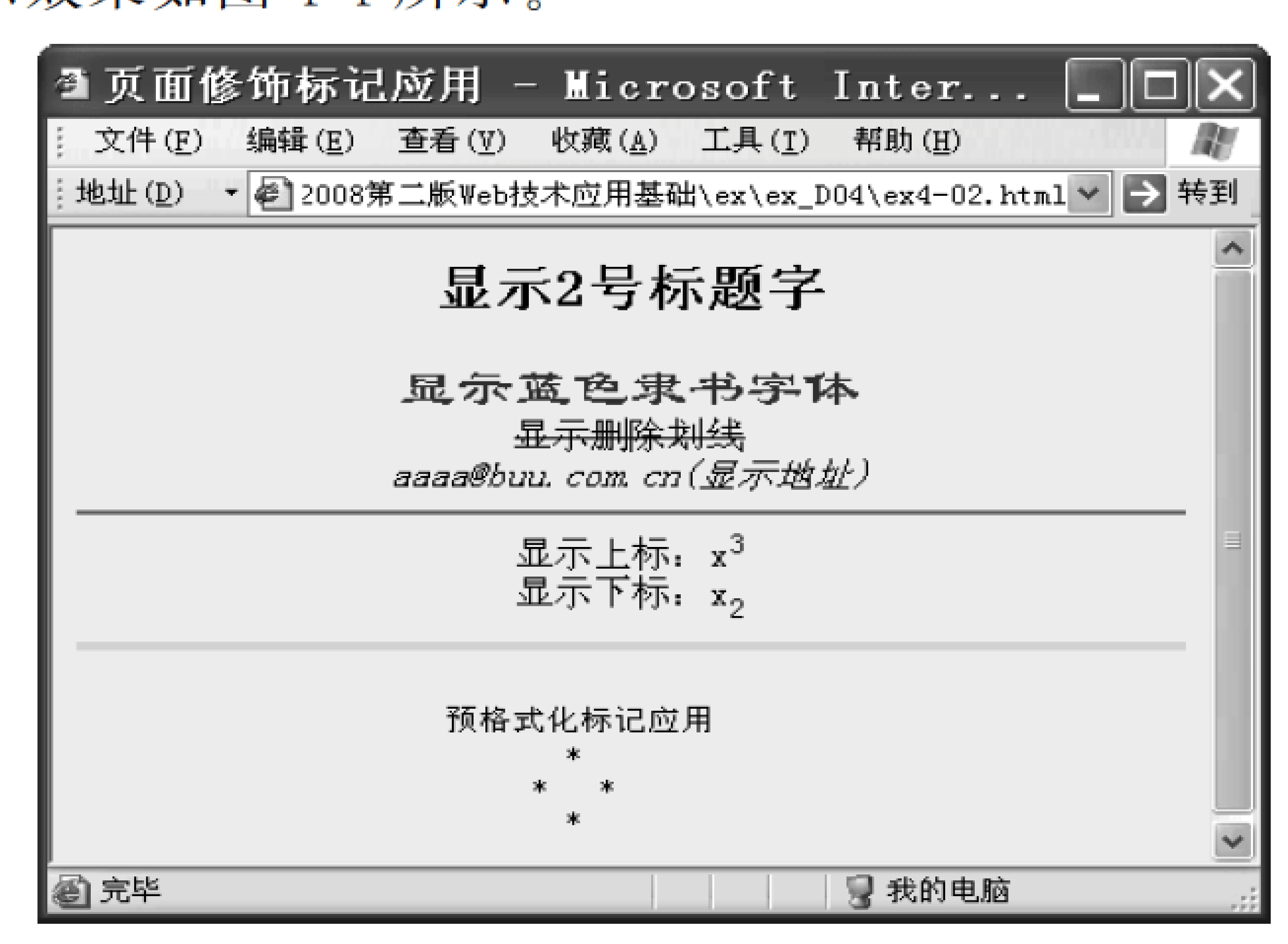


图 4-4 页面修饰标记应用示例

4.3 页面多媒体技术

4.3.1 图像标记<img>

语法：

<img src="URL" alt="文字说明">

图像标记<img>把图像嵌入 HTML 文档中。图像标记的属性见表 4-5。

表 4-5 <img>标记的属性

属 性	功 能	示 例
src	src 是必选项，它指定图像文件的 URL	<img src="ex4-03.jpg">
alt	定义一个文本串，浏览器未完全读入图像或因故不能显示图像时，图像位置显示文本串。浏览器可以显示图像时，alt 属性不起作用	<img alt="新疆喀纳斯湖！，jpg 文件 (2750 bytes)">
align	文本与图像的对齐方式，可取值有：left、middle、right、top 和 bottom	<img align=bottom> 图像底部与文本对齐
border	图像边框宽度(以像素为单位)，border=0 无边框	<img border=3>
width 和 height	图像的高度和宽度(以像素为单位)	<img width="105" height="75">

4.3.2 背景音乐标记<bgsound>

语法：

<bgsound src ="音乐文件的 URL" loop=音乐播放次数>

背景音乐标记<bgsound>可在展示页面的同时播放音乐。当属性 loop=－1 表示音乐将不断地循环播放。

例如：

<bgsound src="music.wav" loop=－1>

4.3.3 音乐和影像文件

语法：

<embed src="音乐或影像文件的 URL" width=宽度 height=高度 autostart=是否自动播放 loop=是否重复>

<embed>标记把音乐和影像嵌入页面，其属性功能如下：

(1) autostart：autostart＝true。true 表示打开页面时自动播放；no 表示不播放。默认值是 no。

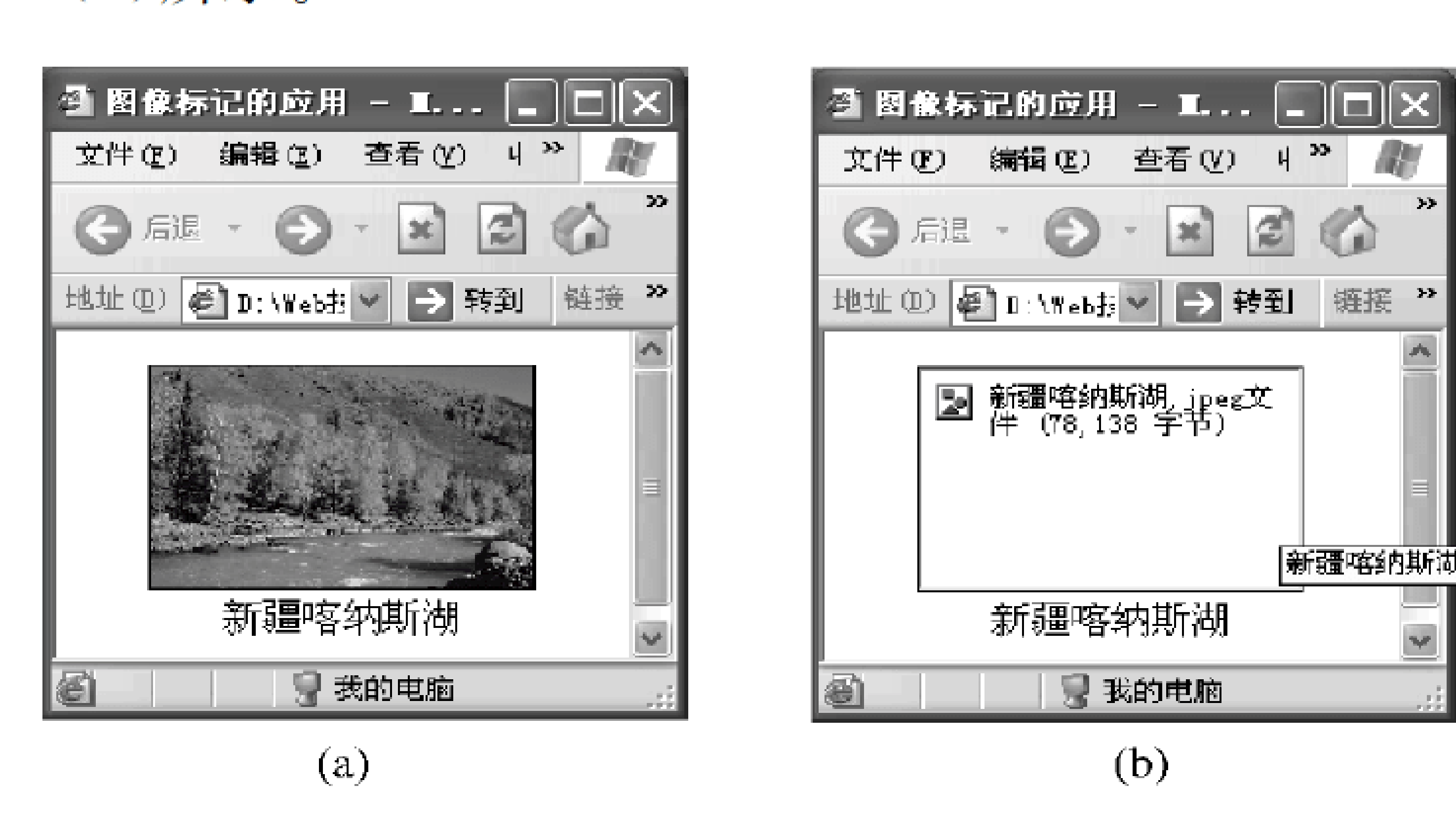
(2) loop：loop＝true。true 表示无限制重复播放；no 表示只播放一次，默认值是 no。

4.3.4 页面多媒体技术应用案例

例 4.3 页面代码 ex4-03.html 说明了图像标记的使用，代码清单如下：

```
<html><head>
    <title>图像标记的应用</title>
</head>
<body>
<img src="ex4-03.jpg" align=bottom border=1 width="153" height="99"
    alt="新疆喀纳斯湖,jpeg 文件 (78,138 字节)">
<br>新疆喀纳斯湖
</body></html>
```

当图像能够在浏览器中显示时，显示如图 4-5(a)所示；当图像因故不能在浏览器中显示时，显示如图 4-5(b)所示。



(a) (b)

图 4-5 图像标记应用案例

例 4.4 页面代码 ex4-04.html 说明了音乐文件的使用，代码清单如下：

```
<html><head>
    <title>自动播放音乐文件</title>
</head>
<body>
<center>
自动播放音乐文件<br>
<embed src="音乐文件.mid" width="307" height="198"
    autostart=true loop=true>
</body></html>
```

注："音乐文件.mid"和页面文件存放在同一目录下。

在浏览器中的运行效果如图 4-6 所示，页面显示时，同时播放"音乐文件.mid"中的音乐。



图 4-6 音乐文件使用案例

例 4.5 页面代码 ex4-05.html 说明了影像文件的使用，代码清单如下：

```
<html><head>
    <title>自动播放影像文件</title>
</head>
<body><center>
自动播放影像文件<br>
<embed src="影像文件.mpg" width="307" height="198"autostart=true loop=true>
</body></html>
```

读者可以在浏览器中测试页面效果。

4.4 表格与列表标记

4.4.1 表格标记<table>…</table>

1. 表格标记的使用格式

HTML 中使用<table>标记建立表格，它的使用格式如下：

```
<table border>
<caption>表格标题</caption>
<tr>
<th>第 1 列表头</th><th>第 2 列表头</th>…<th>第 n 列表头</th>
</tr>
<tr>
<td>第 1 行、1 列表项</td><td>第 1 行、2 列表项</td>…<td>第 1 行、n 列表项</td>
</tr>
<tr>
<td>第 2 行、1 列表项</td><td>第 2 行、2 列表项</td>…<td>第 2 行、n 列表项</td>
</tr>
…
<tr>
<td>第 n 行、1 列表项</td><td>第 n 行、2 列表项</td>…<td>第 n 行、n 列表项</td>
</tr>
</table>
```

表格由<table>、<caption>、<tr>和<td>4 个标记建立，它们的作用如下：

（1）在<table>与</table>之间定义表格标题、表头及单元格中的内容。<table>标记具有属性 border 用于设置边框的宽度，它的取值以像素为单位，例如 1、2、3 等。<table border=2>表示表格带有边框，宽度为 2 个像素。默认值是 border=1，当 border=0 时表格没有边框。

（2）在<caption>与</caption>之间定义表格标题。表格标题具有属性 align，align=top 表格标题位于表首，默认表示标题位于表首；align=bottom 表格标题位于表尾。

（3）每一行以<tr>开始，用</tr>结束。表头元素用<th></th>定义，表头显示成黑体。

（4）单元格内容用<td></td>定义。

2. 表格标记应用案例

例 4.6　文件 ex4-06.html 制作了一个课表，代码清单如下：

```
<html><head>
  <title>表格标记应用案例</title>
</head>
<body>
  <table border=2>
    <caption>0705331 课表</caption>
  <tr>
    <th>节次</th><th>星期一</th><th>星期二</th><th>星期三</th><th>
星期四</th><th>星期五</th>
  </tr>
  <tr>
    <td>1、2</td><td>英语</td><td>操作系统</td><td>网络基础</td>
    <td>英语</td>
    <td>数据库原理</td>
  </tr>
  <tr>
    <td>3、4</td><td>Java</td><td>数据库原理</td><td>实验</td>
    <td>Java</td><td>操作系统</td>
  </tr>
  <tr>
    <td>5、6</td><td>网络基础</td><td>实验</td><td>实验</td>
    <td>实验</td>
  </tr>
</table>
</body></html>
```

在浏览器中的显示效果如图 4-7 所示。



图 4-7　课表

4.4.2　列表标记

在制作网页时，使用列表可以使页面布局清晰明了。HTML 提供了三种类型的列表：
标记定义无序列表(unorder list)；
标记定义有序列表(order list)；<dl>标记定义定义列表(description list)。在列表中的每一项以标记
开始，标记
结束。列表可以嵌套使用，一个列表中的列表项又可以是一个列表。

1. 无序列表标记
…

使用
标记定义一个无序列表，
需要使用尾标记。它具有属性 type，type 可以取以下值：

type=disk：加重符号是实心圆点(默认)。

type=circle：加重符号是空心圆点。

type=square：加重符号是实心方块。

例 4.7　本例说明了无序列表标记的使用，页面代码 ex4-07.html 清单如下：

```
<head>
    <title>无序列表标记应用案例</title>
</head>
<body>
<h4>无序列表 1</h4>
<ul>
  <li><h5>第一项。</h5></li>
  <li><h5>第二项。</h5></li>
</ul>
<h4>无序列表 2</h4>
<ul type=circle>
  <li><h5>第一项。</h5></li>
  <li><h5>第二项。</h5></li>
</ul>
</body></html>
```

该页面代码在浏览器中的显示效果如图 4-8 所示。

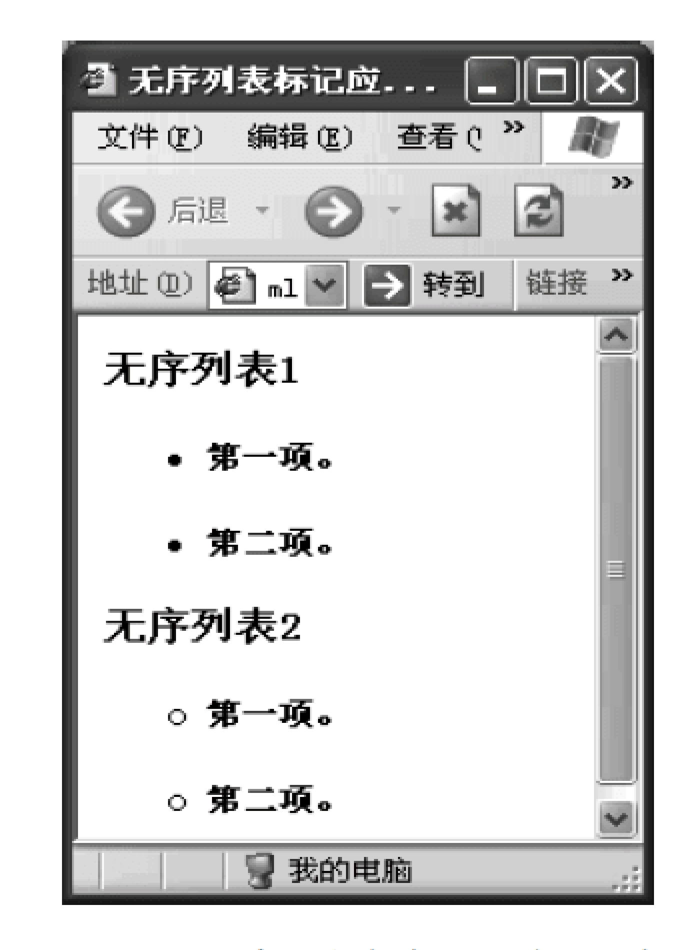


图 4-8 无序列表标记应用案例

2. 有序列表标记
…

使用
标记定义一个有序列表,列表中各项的序列号由浏览器自动给出。
需要使用尾标记。

有序列表标记<ol>有属性 type 和 start。属性 type 可以取以下值。

type=1:默认值,用数字 1、2、3 等标识各项。

type=A:用大写字母 A、B、C 等标识各项。

type=a:用小写字母 a、b、c 等标识各项。

type=I:用大写罗马字母标识各项。

type=i:用小写罗马字母标识各项。

用 start 属性指定列表从哪个数字开头,例如 type=A 的有序列表,如果令 start=3,这个列表的第一项将从 C 开始,以下为 D、E、F 等。

标记可以具有属性 value,它把这一项的列表编号指定为特定值。

例 4.8 本例说明无序列表标记的使用,页面代码 ex4-08.html 清单如下:

```
<html><head>
    <title>有序列表标记应用案例</title>
</head>
<body>
<h3>有序列表的属性</h3>
<ol type=I>
<li><h4>有序列表标记具有属性 start,指定列表编号的开头值。</h4>
<li><h4>有序列表标记具有属性 type</h4>
    <ol type=1 start=6>
    <li><h5>type=1,默认值,用数字 1、2、3 等标识各项。</h5></li>
    <li><h5>type=A,用大写字母 A、B、C 等标识各项。</h5></li>
    <li><h5>type=a,用小写字母 a、b、c 等标识各项。</h5></li>
    <li><h5>type=I,用大写罗马字母标识各项。</h5></li>
    <li value=3><h5>type=i,用小写罗马字母标识各项。</h5></li>
    </ol>
</ol>
</body></html>
```

图 4-9 是该页面代码在浏览器中的显示效果。

3. 定义列表标记<dl>…</dl>

语法:

```
<dl>
```

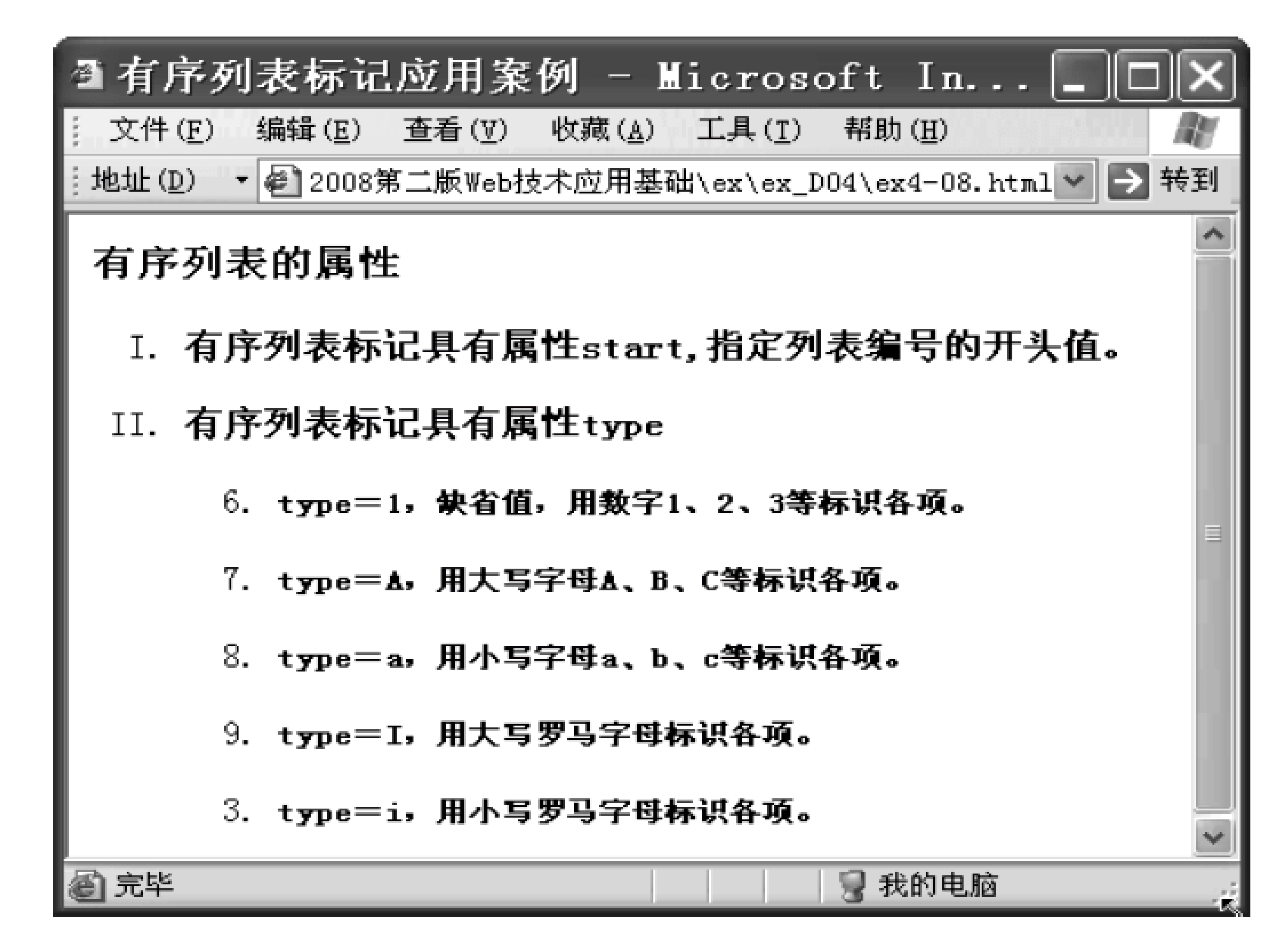


图 4-9　有序列表标记应用案例

```
<dt>术语<dd>术语的定义 1
         <dd>术语的定义 2
<dt>术语<dd>术语的定义
</dl>
```

定义列表的每一项由两部分组成：一部分是术语;另一部分是术语的定义。定义列表由始标记<dl>开始,以尾标记</dl>结束。表中可以有若干个列表项,每个列表项有两个元素：术语用<dt>标记描述;定义用<dd>标记描述。定义列表标记<dl>可以有属性 compact,该属性使术语和它的定义在同一行显示。

例 4.9　本例说明了定义列表标记的使用,ex4-09.html 代码清单如下：

```
<html><head>
    <title>定义标记应用案例</title>
</head>
<body>
<h3>定义列表的使用</h3>
<dl>
  <dt><b>列表</b><dd>数据项的有序集合。
  <dt><b>标号</b><dd>程序中指令的标识符。
                      <dd>磁盘文件的标识说明记录。
                      <dd>数据集中或附加在数据集上的字符,它包括数据集的信息。
</dl>
<dl compact>
  <dt><b>列表</b><dd>数据项的有序集合。
  </dl>
</body></html>
```

图 4-10 是该页面代码在浏览器中的显示效果。

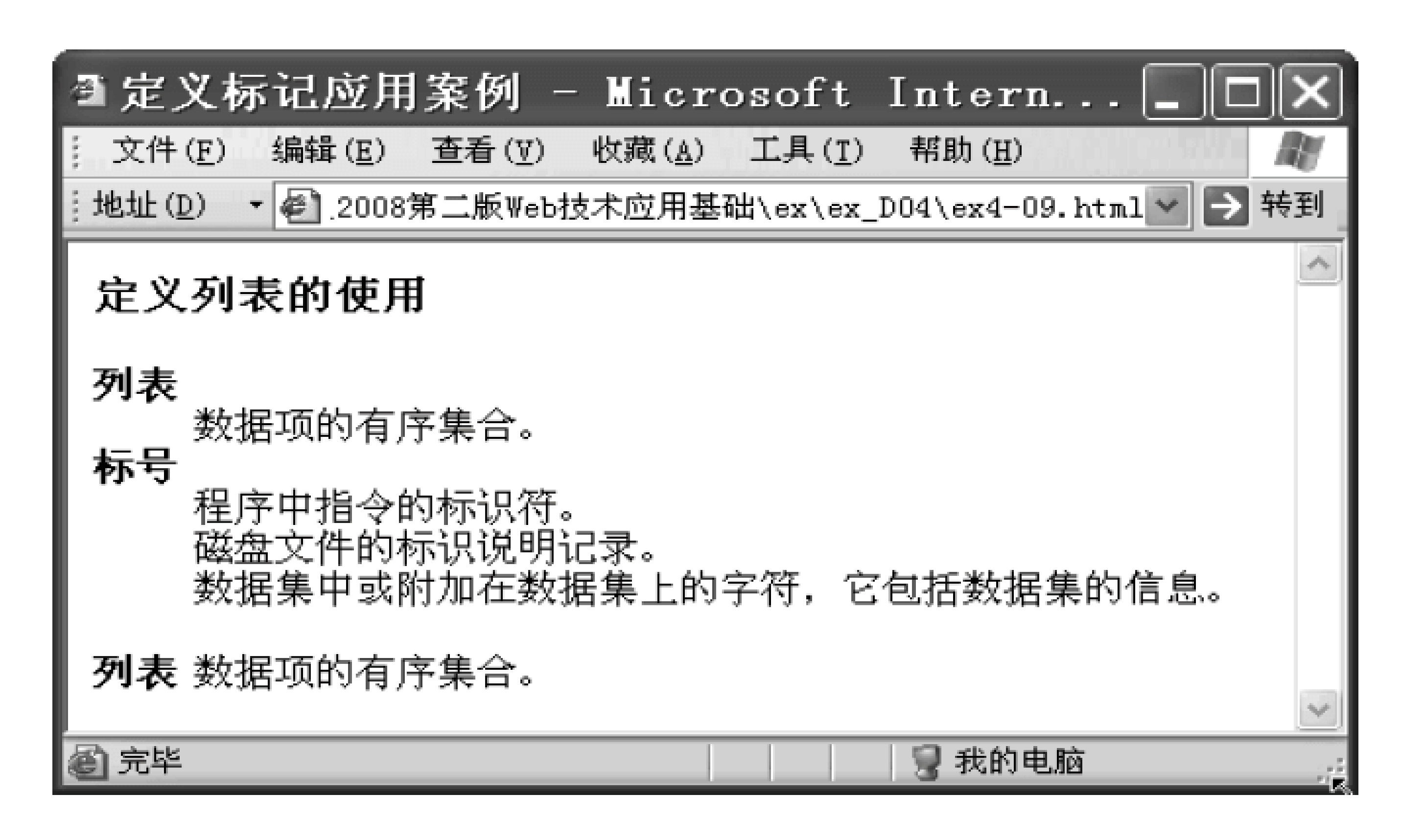


图 4-10　定义列表标记应用案例

4.5　超链接标记

超链接标记是超文本的基本结构，它可以从当前 Web 页面定义的位置跳转到其他位置，包括同一页面的其他位置、Internet 上的另一个 Web 页面、本地硬盘或局域网上的文件、声音或图像文件、FTP 或 Telnet 站点、电子信箱等。

4.5.1　超链接标记<a>…

语法：

超链接显示内容

超链接标记<a>的属性 href 指明所要链接资源的 URL，在标记<a>和之间的内容是链接点，单击链接点网页跳转至 href 指明的资源处。

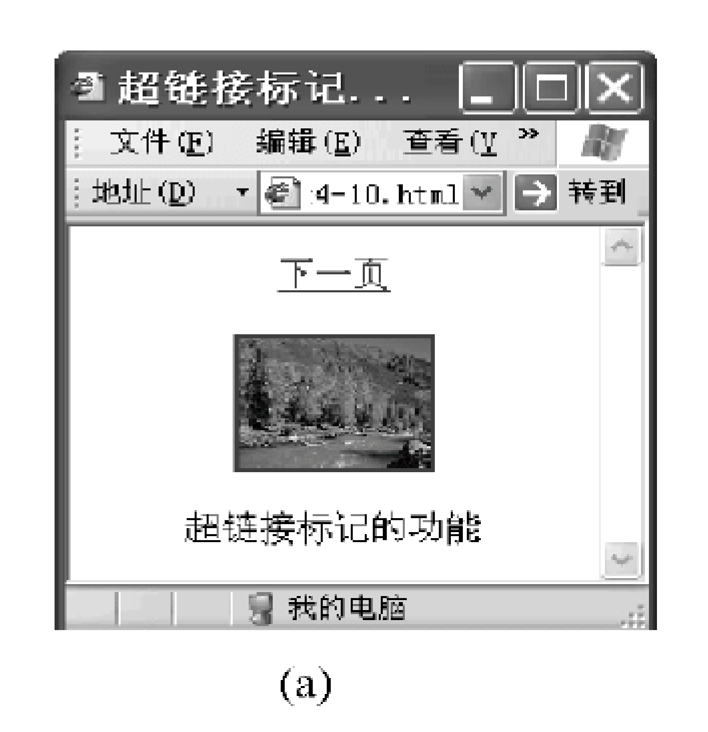
例 4.10　ex4-10.html 和 ex4-10_1.html 演示了不同 Web 页面之间的链接，ex4-10.html 文件代码清单如下：

```
<html><head>
<title>超链接标记应用</title>
</head>
<body><center>
<a href="ex4-10_1.html">下一页</a><p>
<a href="ex4-10_1.html"><img src="ex4-03.jpg" width=82 height=62></a><p>
超链接标记的功能</center>
</body>
</html>
```

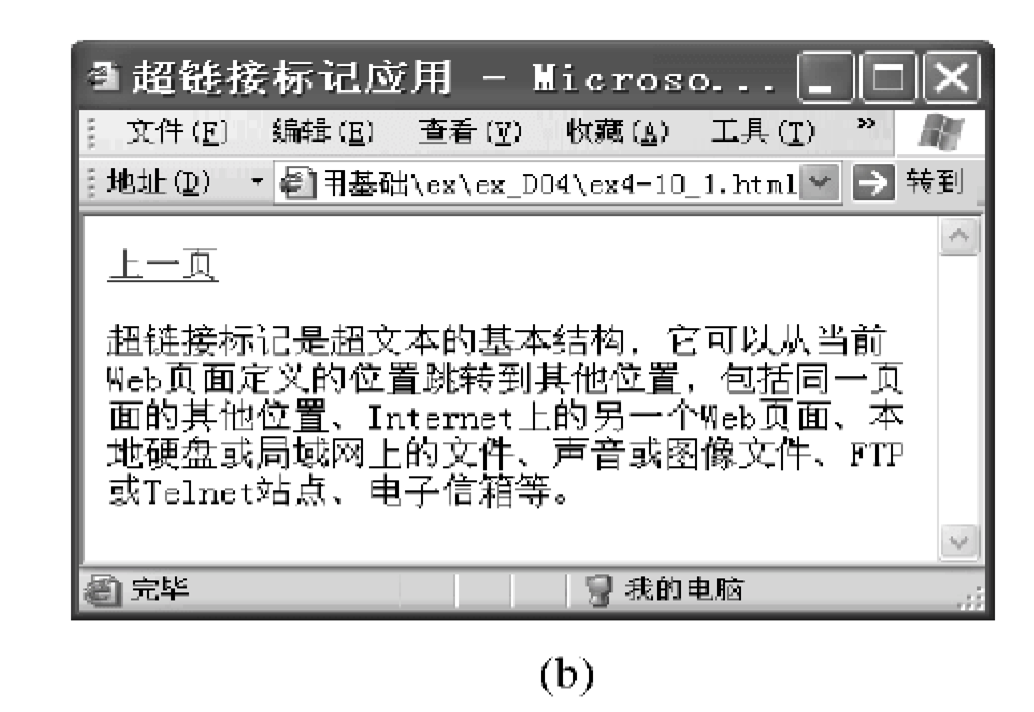
ex4-10_1.html 文件代码清单如下：

```
<html><head>
<title>超链接标记应用</title>
</head>
<body>
<a href="ex4-10.html">上一页</a><p>
超链接标记是超文本的基本结构，它可以从当前 Web 页面定义的位置跳转到其他位置，包括同一页面的其他位置、Internet 上的另一个 Web 页面、本地硬盘或局域网上的文件、声音或图像文件、FTP 或 Telnet 站点、电子信箱等。
</body></html>
```

ex4-10.html 和 ex4-10_1.html 在浏览器中的显示结果见图 4-11，在文件 ex4-10.html 显示的页面上（见图 4-11(a)）单击“下一页”或图像时，即可链接到由 ex4-10_1.html 文件显示的网页（见图 4-11(b)）。



(a)



(b)

图 4-11　超链接应用案例

4.5.2　同一页面间的链接

当 Web 页面比较大跨越几个屏幕时，为阅读方便，可以设置页面内的超链接，相当于提供了一个屏幕滚动功能。语法：

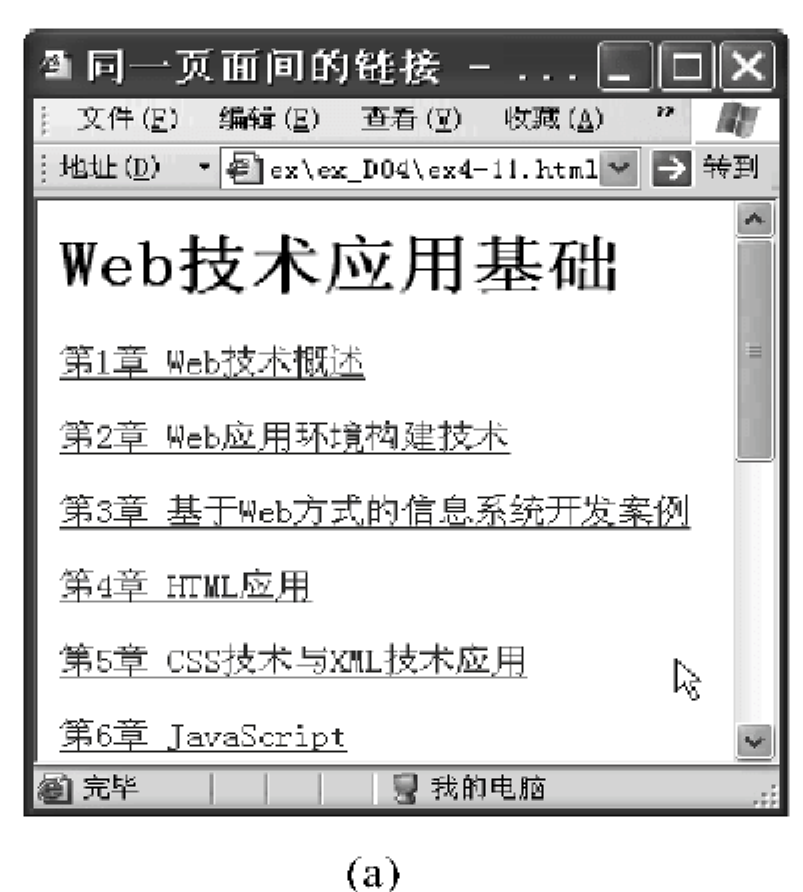
```
<a name="超链接目标名称">目标内容</a>
<a href="#超链接目标名称">超链接显示内容</a>
```

例 4.11　文件 ex4-11.html 说明了同一页面间链接的应用，具体代码如下：

```
<html><head>
    <title>同一页面间的链接</title>
</head>
<body>
<h1>Web 技术应用基础</h1>
<a href="#第 1 章">第 1 章 Web 技术概述</a><p>
```

```
<a href="#第 2 章">第 2 章 Web 应用环境构建技术</a><p>
<a href="#第 3 章">第 3 章 基于 Web 方式的信息系统开发案例 </a><p>
<a name="第 4 章标题" href="#第 4 章" >第 4 章 HTML</a><p>
<a href="#第 5 章">第 5 章 CSS</a><p>
<a href="#第 6 章">第 6 章 XML<p>
<a href="#第 7 章">第 7 章 JavaScript</a><p>
<a href="#第 8 章">第 8 章 Web 数据库编程技术</a><p>
<a href="#第 9 章">第 9 章 JSP 技术应用</a><p>
<a href="#第 10 章">第 10 章 ASP 技术应用</a><p>
<a href="#第 11 章">第 11 章 综合实训</a><p>
<a name="第 4 章" href="#第 4 章标题">第 4 章 HTML</a><br>
    4.1 HTML 概述<br>
    4.2 HTML 页面修饰标记<br>
    4.3 页面多媒体技术<br>
    4.4 表格与列表标记<br>
    4.5 超链接标记<br>
    4.6 表单标记<br>
    4.7 窗口框架标记<br>
</body></html>
```

页面显示结果如图 4-12 所示。





(a) (b)

图 4-12　同一页面间超链接应用案例

单击图 4-12(a)的“第 4 章 HTML 应用”文本，即可跳转到网页中具有目标名第 4 章 HTML标记的位置，如图 4-12(b)所示。

4.5.3　链接电子信箱

语法：

链接文本

例如，下面语句建立了一个电子信箱的超链接，单击“联系我们”即可发信：

```
<a href="mailto:aaaa@buu.com.cn"> 联系我们 </a>
```

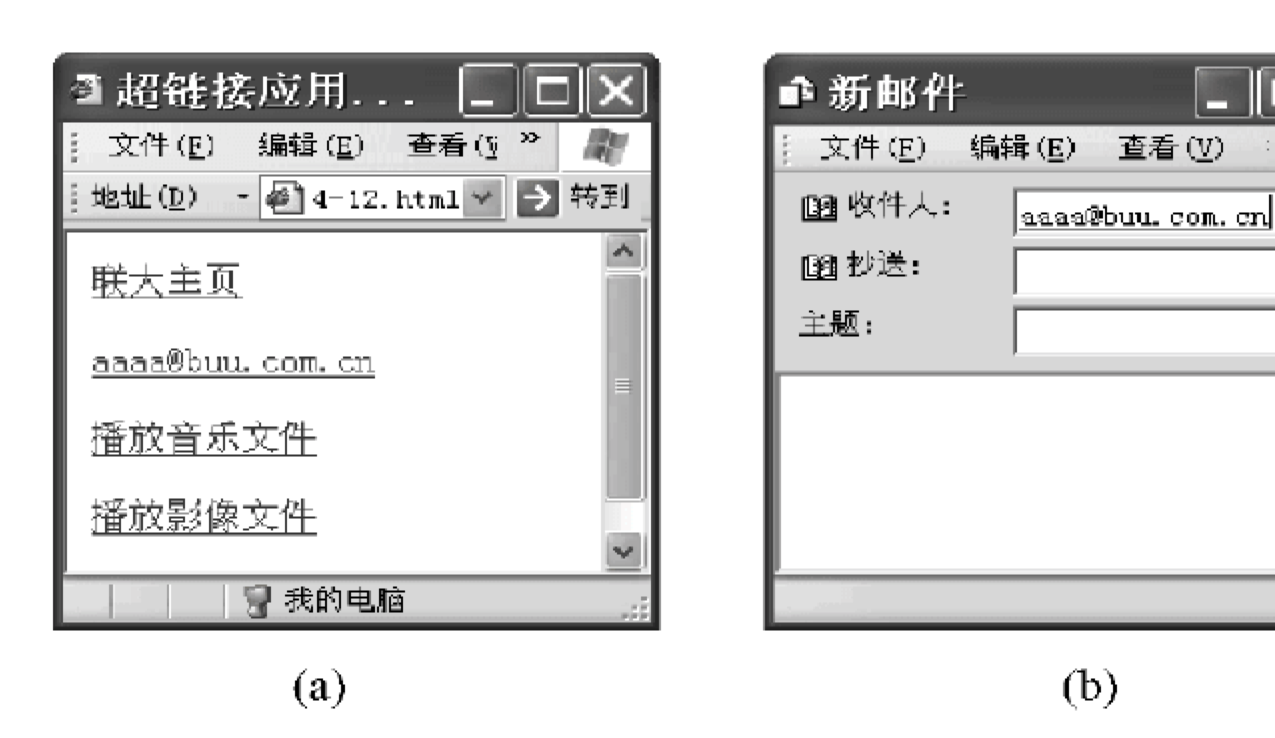
4.5.4 超链接应用案例

应用超链接可以应用多种协议链接到其他网站，也可以链接到其他文件，例如音乐和影像文件。

例 4.12 文件 ex4-12.html 上有四个超链接，分别链接到北京联大主页、电子信箱、播放音乐或影像文件，具体代码如下：

```
<html><head>
    <title>超链接应用案例</title>
</head>
<body>
<a href="http://www.buu.edu.com">联大主页</a><p>
<a href="mailto:aaaa@buu.com.cn"> aaaa@buu.com.cn </a><p>
<a href="音乐文件.mid">播放音乐文件</a><p>
<a href="影像文件.mpg">播放影像文件</a><p>
</body></html>
```

文件 ex4-12.html 在浏览器中的显示效果如图 4-13(a)所示，依次单击页面上的超链接，将分别链接到北京联大主页、电子信箱、播放音乐文件或影像文件，图 4-13(b)显示链接电子信箱。文件 ex4-12.html 中的联大主页，表示使用 http 协议链接到 www.buu.edu.com 网页； aaaa@buu.com.cn 表示使用 mailto 发送电子邮件到 aaaa@buu.com.cn 信箱。



(a) (b)

图 4-13 超链接应用案例

4.6 表 单 标 记

4.6.1 表单的功能

<form>标记在页面中产生表单，表单提供了用户与 Web 服务器的信息交互功能，是动态 Web 技术的要素之一。表单接收用户信息，并把信息提交给服务器，然后由服务器端的应用程序处理信息，把处理结果返给用户并向用户显示。通过交互性，可以在网上完成注册个人信息、检索信息和购物等活动。

表单的定义标记是<form>，表单中包含的三个基本控制标记是<input>、<select>和<textarea>。

4.6.2 表单定义标记<form>…</form>

语法：

<form method="post|get" action="URL" enctype ="type"> </form>

<form>标记具有三个属性。

- action：用来指定完成表单信息处理任务服务器程序的完整 URL，例如 http://www.buu.edu.cn/bin/ReaderNote.jsp。
- method：指定表单中输入数据的传输方法，它的取值是 get 或 post，默认值是 get。当 method=get 时，表示将表单中输入数据打包后加到在 action 中指定的 URL 后面。post 与 get 的功能相似，只是 post 将表单中的数据和调用程序分开发送，一般来讲使用 post 比较安全。
- enctype：指定表单中输入数据的编码方法。在 method=post 情况下才有效。

4.6.3 输入标记<input>

<input>标记定义输入控件，它的类型由 type 属性来确定。

语法：

<input type=控件类型　name=控件名称 … >

<input>标记的属性如下。

- name：定义控件的标识。
- value：提供控件输入域的初始值。
- maxlength：定义控件输入域中允许输入的最多字符数。
- size：定义控件输入域的大小。
- checked：提供复选框和单选按钮的初始状态。

- URL 和 align：指定在图像按钮中使用的图像所在位置（URL）和图像的对齐方式。
- type：用以指定控件的类型，例如 text（文本框）、radio（单选按钮）等，默认值是 text。属性 type 的取值及它们的意义见表 4-6。

表 4-6　type 组件类型

type 组件类型	组件名称及作用
button	按钮，可以创建提交按钮、复位按钮和普通按钮
text	文本框，接收任何形式的输入，如字母、数字等
password	口令框，用“·”号代替输入字符的显示
checkbox	复选框，提供多项选择
radio	单选按钮，提供单项选择
submit	提交按钮，单击提交按钮，发送表单信息提交服务器
image	图像按钮，单击图像，发送表单信息提交服务器
reset	复位按钮，把表单上的组件值恢复为默认值
hidden	隐藏表单组件，把表单中一个或多个组件隐藏起来
textarea	多行文本框
file	上载文件

4.6.4　列表框标记<select>…</select>

语法：

```
<select  name="下拉列表框名称" size="下拉列表框显示的条数" >
    <option value="控件的初始值" selected > 选项描述
    <option value="控件的初始值"          > 选项描述
    …
</select>
```

使用<select>标记定义下拉式列表框和滚动式列表框，<select>标记具有三个属性。

- name：指定列表框的名字。
- size：定义列表框显示的条数，也就是一次可以看见的列表项的数目。
- multiple：允许进行多项选择。

使用<select>标记定义列表框时，由<option>标记定义列表框的各个选项。<option>标记具有以下属性。

- selected：表示该项预先选定。

- value：指定控件的初始值。

4.6.5 多行文本框标记<textarea>…</textarea>

语法：

<textarea cols＝文本宽度 rows＝文本区行数>…</textarea>

<textarea>标记的作用与文本框类似，用来输入文本信息，其中 cols 属性以字符为单位。

4.6.6 表单标记应用案例

例 4.13 ex4-13.html 说明了表单标记的使用，代码清单如下：

```
<html><head>
    <title>表单标记应用案例</title>
</head>
<body>
<form method="post" action="travel.asp">
请输入姓名：<input type=text name="txtname" size=12 maxlength=6><p>
请输入密码：<input type=password name="pasname" size=12 maxlength=6><p>
上载的文件：<input type=file name="filename" size=12 maxlength=6><p>
性别：<select  name="性别">
          <option> 男
          <option> 女
      </select><p>
请选择旅游城市，可做多项选择
<input type=checkbox name="复选框 1" checked> 北京
<input type=checkbox name="复选框 2" > 上海
<input type=checkbox name="复选框 3" > 西安
<input type=checkbox name="复选框 4" > 昆明<p>
请选择付款方式
<input type=radio name="单选按钮 1" > 信用卡
<input type=radio name="单选按钮 1" checked> 现金<p>
留言：<p>
<textarea cols=50 rows=4>请与我们联系</textarea><p>
<input type=reset name="复位按钮" value="复位">
<input type=submit name="提交按钮" value="确定">
</form>
</body></html>
```

页面在浏览器中的显示效果如图 4-14 所示。

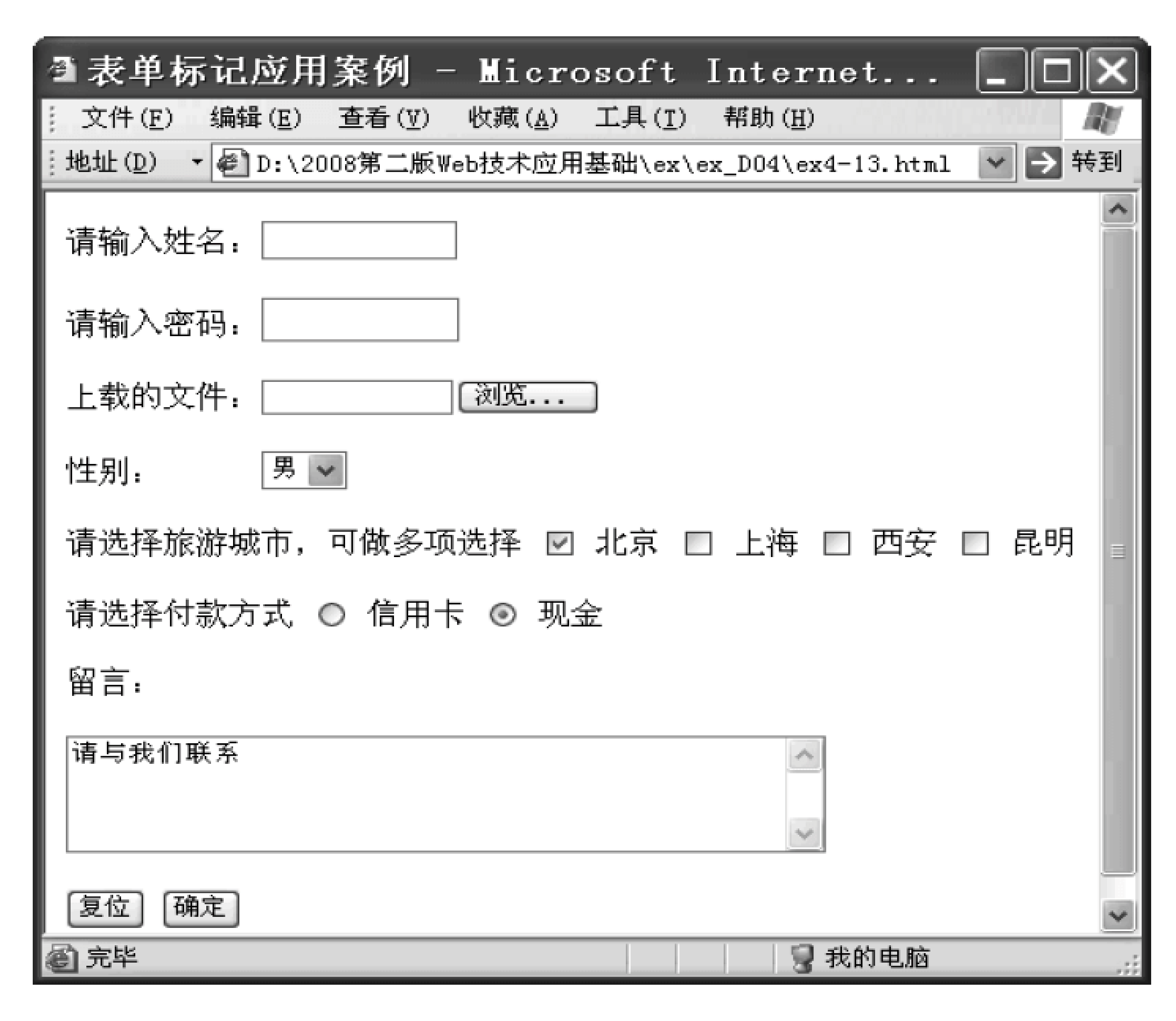


图 4-14　表单标记应用案例

4.7　窗口框架标记

使用窗口框架可以把浏览器窗口划分成几个大小不同的子窗口，每个子窗口显示不同的页面，可以在同一时间浏览不同的页面。

4.7.1　窗口框架的建立

在建立窗口框架时，先要定义一个框架集，然后逐个定义每个窗口。

语法：

＜frameset rows="行高列表" cols="列宽列表" frameborder=0|1 border=n bordercolor=颜色＞…＜/frameset＞

＜frameset＞属性应用如下。

- rows：说明窗口水平分割情况，"行高列表"是一组用","分开的数值，可以用百分数来表示，也可以用像素表示，指定了各子窗口的高度。
- cols：说明窗口垂直分割情况，"列宽列表"也是一组用","分开的数，定义了各子窗口的宽度。
- frameborder：属性指定框架中所有子窗口是否显示边框，0 是不显示边框，1 是显示边框。
- border：指定分割窗口框架的宽度，以像素为单位。

- bordercolor：指定框架的颜色。

4.7.2 子窗口的建立

<frame>标记定义子窗口。

语法：

<frame src="子窗口显示文件的 URL" name="子窗口名称">…</frame>

4.7.3 窗口框架使用案例

例 4.14 本例说明了窗口标记的使用。

(1) 框架集 ex4-14.html 的代码清单如下：

```
<html><head>
    <title>窗口框架应用案例</title>
</head>
<frameset rows="30%,*" border=5>
     <frame src="top.html" name="top">
     <frameset cols="30%,*">
         <frame src="left.html" name="left">
         <frame src="right.html" name="main">
     </frameset>
</frameset>
</html>
```

(2) 顶层子窗口 top.html 的代码清单如下：

```
<html><head>
    <title>网上书店</title></head>
<body><center>
<font face="华文彩云" size=7 color=blue><i>网 上 书 店</i></center>
</body></html>
```

(3) 左边子窗口 left.html 的代码清单如下：

```
<html><head>
    <title>网上书店</title></head>
<body><center>
<a href="ex4-02.html" target="main">页面修饰标记的应用</a><p>
<a href="ex4-08.html" target="main">无序列表标记的应用</a><p></center>
</body>
</html>
```

其中超链接标记<a>的 target 属性指明链接到的子窗口的名称。
(4) 右边子窗口设置了一幅背景图片,right. html 的代码清单如下:

```
<html><head>
  <title>网上书店</title></head>
<body background="back1.gif">
</body>
</html>
```

页面在浏览器中的显示效果如图 4-15 所示。

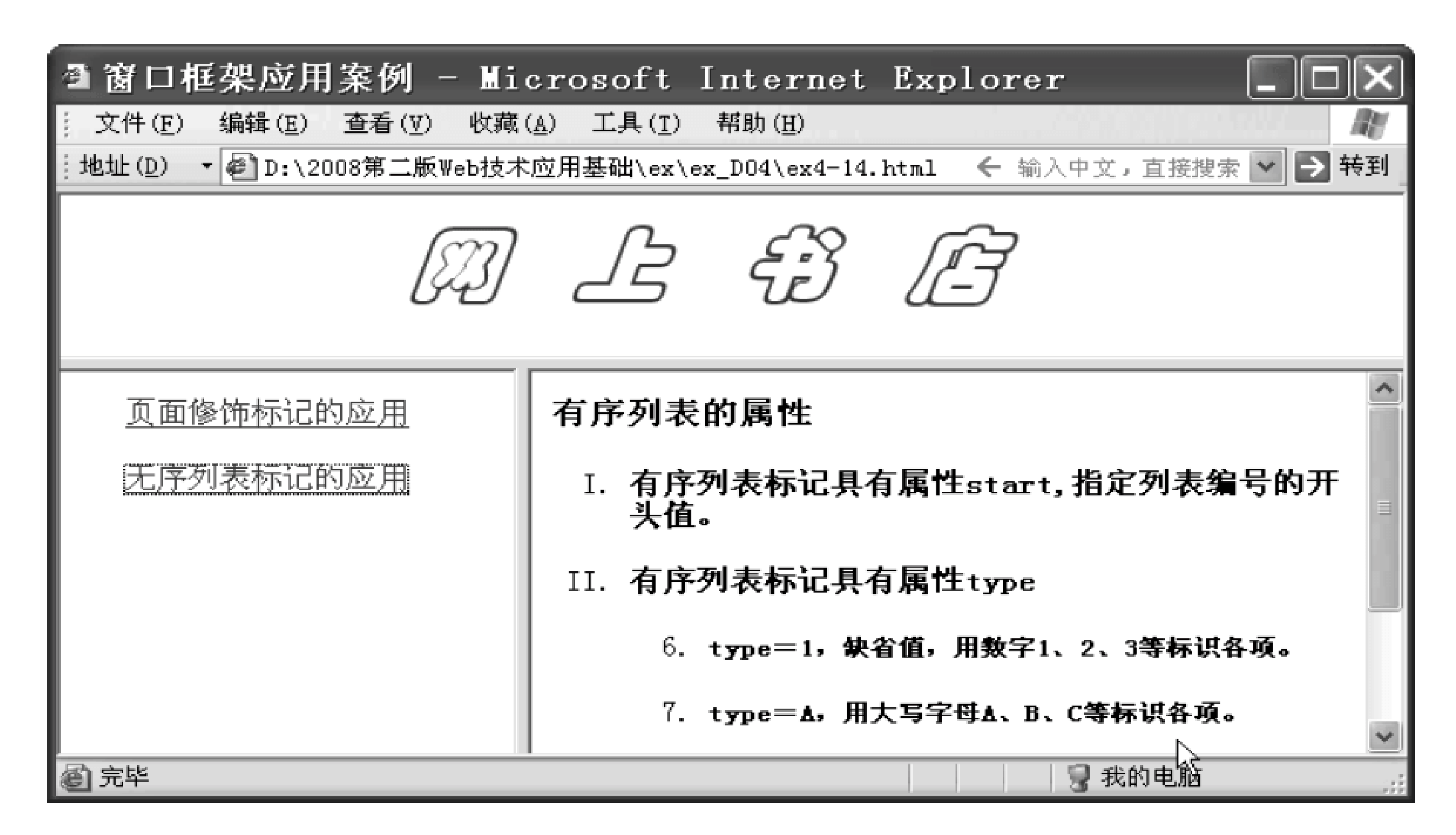


图 4-15 窗口框架应用案例

4.8 HTML 应用案例

4.8.1 页面动态刷新

页面动态刷新功能使一个页面停留几秒钟后,自动跳转到另一个页面。
语法:

<meta http-equiv="refresh" content="秒数;URL=目标网址">

例 4.15 说明页面动态刷新功能的 ex4-15. html 代码清单如下:

```
<html><head>
    <meta http-equiv="refresh" content=6;URL=ex4-14.html>
    <title>页面动态刷新功能案例</title>
</head>
<body>
```

```
6 秒钟后自动跳转到"网上书店"页面
</body></html>
```

读者可以自行在浏览器中测试该页面的显示效果。

4.8.2 文字移动

在页面中显示移动的文字,也称走马灯。

语法:

<marquee height=n width=n direction=left | right | up | down behavior=scroll | slide | alternate bgcolor=颜色>

<marquee>标记的属性功能如下。

(1) height:走马灯的高度,以像素为单位。

(2) width:走马灯的宽度,以像素为单位。

(3) direction:文字移动方向。

(4) behavior:控制文字移动方式。

scroll:文字从右向左移动。

slide:文字从左向右移动。

alternate:移动的文字在走马灯范围内来回移动。

(5) bgcolor:走马灯的背景颜色。

例 4.16 说明文字移动功能的文件 ex4-16.html 代码清单如下:

```
<html><head>
    <title>文字移动功能案例</title>
</head>
<body><center>
  <marquee height=50 width=400 direction=right behavior=alternate bgcolor=#ddffff>
  <font face="隶书" size=5 color=red>Web 技术应用基础</font>
  </marquee></center>
</body></html>
```

ex4-16.html 代码在浏览器中的显示效果如图 4-16 所示,文字"Web 技术应用基础"在屏幕上移动显示。

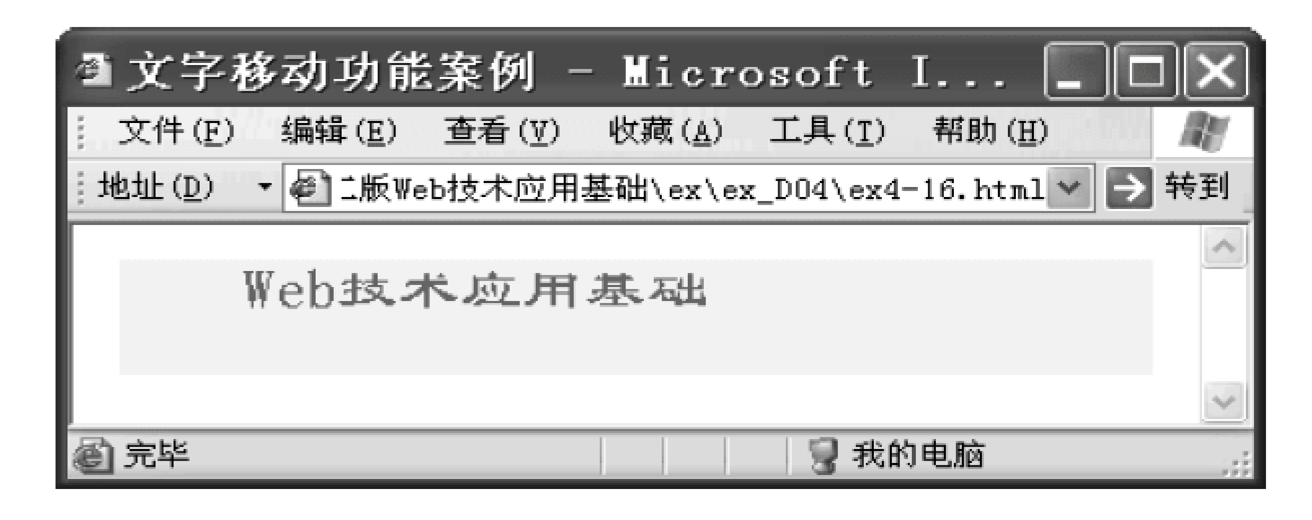


图 4-16 文字移动

4.8.3 浮动窗口

在页面内创建一个浮动窗口。

语法：

```
<iframe>
    有关浏览器不支持浮动窗口的文字说明
</iframe>
```

<iframe>标记在页面内部创建一个浮动窗口，使用方法与<frame>标记类似。

例 4.17 说明浮动窗口功能的文件 ex4-17.html 代码清单如下：

```
<html><head>
    <title>浮动窗口应用案例</title>
</head>
<body><center>
  <iframe name="float">
     浮动窗口
  </iframe><p>
  <a href="ex4-02.html" target="float">页面修饰标记的应用</a><p>
  <a href="ex4-08.html" target="float">无序列表标记的应用</a><p></center>
</body></html>
```

ex4-17.html 代码在浏览器中的显示效果如图 4-17 所示。



图 4-17 浮动窗口

4.8.4 在页面中嵌入 Java 小程序

Java 小应用程序是用 Java 语言编写的一段代码，它被嵌入 HTML 页面中，在浏览器环境中运行。

语法：

<applet code=Applet 文件名称 width=宽度 height=高度>…</applet>

例 4.18 说明<applet>标记使用方法的文件 ex4-18.html 代码清单如下：

```
<html><head>
    <title>在页面中嵌入 Java 小程序</title>
</head>
<body>
<hr>
<applet code=RollingMessage.class width=500 height=80>
</applet>
<hr>
</body></html>
```

语法说明：在 ex4-18.html 代码中嵌入 Java 小程序 RollingMessage.class，RollingMessage.class 是由 RollingMessage.java 生成的字节码文件。RollingMessage 程序的功能是动态显示文字，使文字"欢迎学习 Web 技术应用基础!"一个字一个字地显示。ex4-18.html 代码在浏览器中的显示效果如图 4-18 所示。



图 4-18 在页面中嵌入 Java 小程序

4.9 网上书店主界面的实现

网上书店的主界面见第 3 章图 3-5，文件名为 index.jsp。

把其中的框架结构代码提取出来，代码段如下：

```
<body>
<div align="center">
  <table width="750" border="0" cellspacing="1" cellpadding="1">
    <tr>
      <td colspan="2">
        <div align="center">
          <%@include file="top.jsp" %><! --显示 top.jsp-->
```

```
        </div></td>
    </tr>
    <tr>
      <td width="25%" valign="top">
        <%@include file="declare.jsp" %> <! --显示公告栏-->
        <%@include file="login.jsp" %>    <! --显示用户登录-->
        <%@include file="search.jsp" %>   <! --显示书目搜索-->
      </td>
      <td width="75%" valign="top">
        <! --显示精品图书-->
        <! --显示新书架-->
      </td>
    </tr>
    <tr>
      <td colspan="2">
        <%@include file="bottom.jsp" %><! --显示 bottom.jsp-->
      </td>
    </tr>
  </table>
</div>
</body>
```

习题、上机练习与实训 4

一、习题

1. HTML 的中文名称和英文名称是什么？它在页面中起什么作用？
2. HTML 标记是否区分大小写？可以嵌套使用吗？
3. HTML 文档的扩展名是什么？
4. HTML 的本质是什么？它是什么格式的文件？
5. HTML 文件以什么标记开始？什么标记结束？用什么标记把文档分为两部分？
6. 在 HTML 中，标记<font>的 size 属性的最大值和最小值是多少？
7. 在 HTML 文档中页面背景色和字体颜色如何表示？
8. 在 HTML 文档中<p>标记和
标记的区别是什么？
9. 预格式化标记<pre>的作用是什么？
10. 图像标记的 alt 属性起什么作用？
11. 表单在页面中起什么作用？它包含哪些元素？
12. 超链接标记的作用是什么？如何应用超链接标记链接到其他网站、其他页面、电子信箱、音乐和影像文件？

二、上机练习

1. 应用 HTML 制作第一个页面，页面上有一幅图像和有关图像的文字说明。

2. 使用 HTML 制作一个页面，要有背景颜色，页面显示内容及格式如图 4-19 所示。

3. 制作一个页面，有背景音乐，上有文字、图像和朋友的 E-mail 地址，在文字和图像上创建超级链接，单击链接时，页面跳转到其他页面，单击 E-mail 地址时，打开信箱，发送邮件。

4. 制作一个本学期使用的课表。

5. 制作一个计算器界面，界面如图 4-20 所示。

显示粗体字
显示斜体字
混用格式，显示斜、粗体字
显示红色字体

<u>显示下划线</u>
~~删除划线~~

显示上标：X^3
显示下标：X_2

显示大字体
显示小字体

图 4-19 页面显示内容及格式

计算器

请输入第一个操作数:

请输入第二个操作数:

请选择操作
○+ ⊙- ○* ○/

=

结果是:

图 4-20 计算器界面

6. 制作一个框架结构的页面，样式如图 4-21 所示，页面主题自定。

Top：显示 logo 图片和页面的标题。

Left：显示与主题相关的栏目，单击栏目超链接，在 Body 栏显示相关内容。

Body：当前页面的主要显示区域。

Bottom：显示版权信息。

7. 自定义一个主题，应具有以下基本功能：

(1) 用记事本写一个简单的策划报告。

(2) 首页。

① 自己定义一个网站标题。

② 首页框架如图 4-22 所示，每框分别显示不同的内容。

Top
Left
Body
Bottom

图 4-21 框架结构

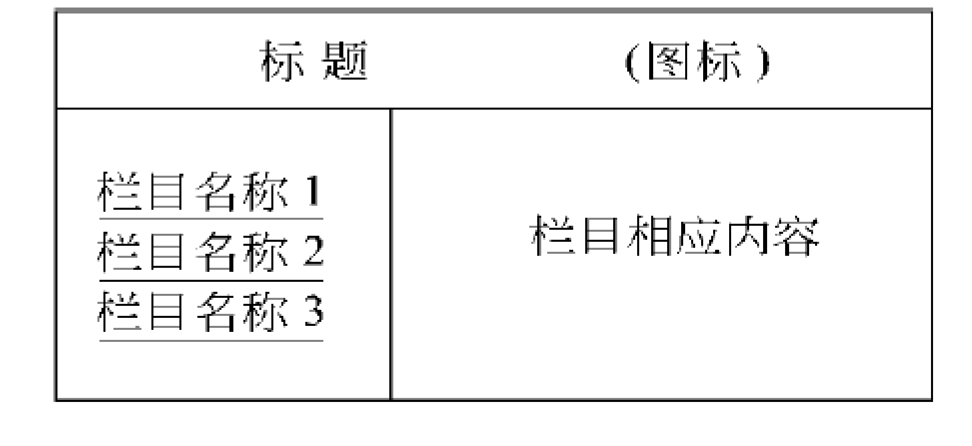


图 4-22 网站首页框架

③ 在左框架显示栏目名称，双击栏目名称，右框架显示相应内容。

④ 采用适当的色彩、背景制作。

⑤ 二级页面有返回主页的链接。

三、实训课题

1. 自学 Dreamweaver、Flash、FrontPage 等软件，制作一个在人才市场推销自己的个

人主页，并在 Web 服务器上发布。

2. 开发一个年级网站，该网站应有：

(1) 你所在系的概貌，并介绍相关信息。

(2) 年级概貌、所学专业，并介绍相关信息。

(3) 你所在班级的概貌，并介绍相关信息。

(4) 每个人的情况介绍。

为完成实训课题 2，需成立：

(1) 年级设计与规划小组，负责年级网站的规划、设计、信息采集与年级网站蓝图的制作，并分配与协调班级网页的制作工作。

(2) 班级设计与规划小组，负责班级网站的规划、设计、信息采集与班级网站蓝图的制作，并分配和协调个人网页的制作工作。

第5章 CSS

通过第4章的学习,读者可以了解到HTML文件中同时包含了Web页面显示内容和显示样式,这将给网站的开发与维护带来一定的难度。人们希望将页面显示内容与显示的方式分离开,可以使内容编辑人员的工作重心放在网站内容的撰写上,而站点的美工和界面设计人员的工作重心放在页面显示的风格和样式上。CSS(Cascading Style Sheets,级联样式表)的工作重点是页面显示的样式。

5.1 CSS简介

CSS是开放性样式设定语句,它扩充了HTML标记的属性设定,这些属性设定可以通过脚本语言进行控制,使网页的视觉效果更加丰富多彩。CSS将页面的样式与显示内容的HTML文件分离开,可以高效、统一地组织页面元素。对于需要具有相同风格多个页面的站点来说,只需要建立定义样式的CSS文件,并使需要这些样式的HTML文件调用相应的样式文件即可。当网站的风格需要更新时,只要更改CSS样式文件就可以了。CSS技术的应用使得网站的开发、管理与维护大为简化,提高了开发效率。

5.1.1 CSS作用

CSS是开放性样式设定语句,它扩充了HTML标记的属性设定,这些属性设定可以通过脚本语言进行控制。CSS基于HTML,它的基本语法仍然是HTML。使用CSS就是将页面样式的定义与HTML文件分离开,建立定义样式的CSS文件后,可由HTML文档调用该样式文件,并按该样式显示。

CSS用于定义Web页面内容在浏览器中的显示方式,由于HTML的功能有限,一般不能随意设计版面和编排文字,所以W3C公布了一套HTML的扩展标准CSS,扩展了HTML在排版和文字样式上的功能。通过样式定义可以设定很多属性,如字号、颜色、页边距、元素在页面上的绝对位置等。

在HTML网页的代码中,网页要展示的样式是在标记内设定的,如:

<h2><font color=red>你好！</h2>

“你好!”显示成红色,是由标记<font>内的属性 color 设定的,样式分散在各个标记中,所以在更改样式时,需要逐个修改各个标记中的属性。

CSS 的概念是：把网页展现的样式从网页中独立出来集中管理,如果需要改变网页样式,只需要改变样式设定部分,HTML 文件本身不需要更改。

5.1.2 CSS 样式文件应用结构

页面的样式设定与页面内容分离后,可以把 CSS 样式信息存成独立的文件,使多个网页文件共享该样式文件。也可以把样式分类,分存于不同的文件,如分为编排样式文件、字体样式文件、颜色样式文件等,把多个样式文件套用在一个网页文件上。样式文件的应用方式见图 5-1。

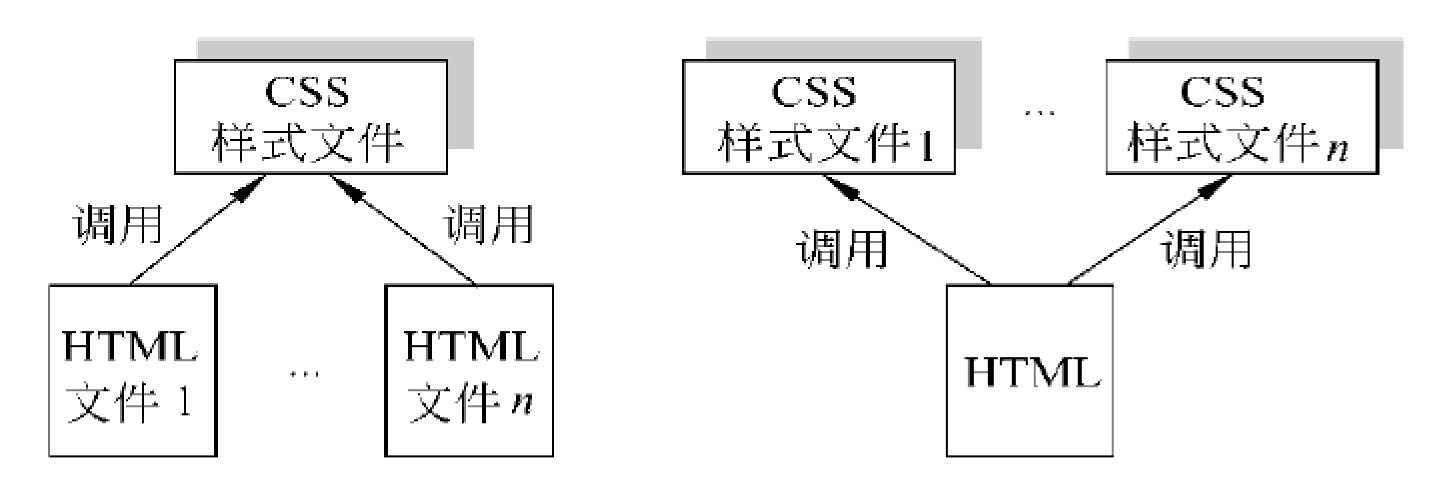


图 5-1 样式文件应用结构

应用样式文件具有以下优点：

① 简化 Web 页面的格式代码。

② 修改样式文件可以更新整个 Web 站点的显示风格,减少网站管理与维护的工作量。

5.2 定义样式的格式

5.2.1 CSS 定义

定义样式的基本格式是：

选择符{ 规则 }

例如：h1{color:blue;}

选择符(Selector)：是样式要套用的对象,一般是 HTML 标记,如上例的 h1。在 HTML 文件中<h1>…</h1>标记之间的内容将全部继承 h1 的全部规则,例如字体显示为蓝色。

规则(Rule)：是样式设定的内容,是用{}括起来的部分,如上例的{color: blue;}。

5.2.2 CSS 属性

CSS 支持的文字样式属性见表 5-1。

表 5-1 文字样式属性

<table>
<tr><th>属　性</th><th>功能与属性值</th><th>示　例</th></tr>
<tr><td>font-family</td><td>定义字体</td><td>font-family:隶书</td></tr>
<tr><td>font-size</td><td>定义字体大小
• 绝对大小：
xx-small|x-small|small|medium|large|x-large|xx-large
• 相对大小：large|small</td><td>font-size:x-large
font-size:60px
font-size:160%
font-size:large</td></tr>
<tr><td>font-weight</td><td>字体粗细
normal|bold|bolder|lighter|100-900</td><td>font-weight:bold</td></tr>
<tr><td>font-variant</td><td>字体变形
normal(普通)|small-caps(小型大写字母)</td><td>font-variant: normal</td></tr>
<tr><td>font-style</td><td>字体效果
normal|italic|oblique</td><td>font-style: italic</td></tr>
</table>

CSS 支持的颜色和背景属性见表 5-2。

表 5-2 颜色和背景属性

<table>
<tr><th>属　性</th><th>功能与属性值</th><th>示　例</th></tr>
<tr><td>color</td><td>定义前景色</td><td>color:red</td></tr>
<tr><td>background-color</td><td>定义背景色
颜色|transparent(透明)</td><td>background-color:white</td></tr>
<tr><td>background-image</td><td>定义背景图像的 URL</td><td>background-image:http://www.buu.com.cn/img/fen.gif</td></tr>
</table>

CSS 支持的长度单位见表 5-3。

表 5-3 CSS 支持的长度单位

<table>
<tr><th colspan="2">类　型</th><th>说　明</th><th>示　例</th></tr>
<tr><td rowspan="2">相对</td><td>em</td><td>相对字符高度,以当前元素本身的 font-size 为参考依据</td><td>margin:4em</td></tr>
<tr><td>px</td><td>以像素为单位</td><td>font-size:16px</td></tr>
<tr><td rowspan="5">绝对</td><td>in</td><td>英寸 1 in=2.54cm</td><td>font-size:0.6in</td></tr>
<tr><td>cm</td><td>厘米</td><td>font-size:0.6cm</td></tr>
<tr><td>mm</td><td>毫米</td><td>font-size:6mm</td></tr>
<tr><td>pt</td><td>点数 1 pt=1/72 in</td><td>font-size:40pt</td></tr>
<tr><td>pc</td><td>印刷单位 1 pc=12 pt</td><td>font-size:4pc</td></tr>
</table>

CSS 提供光标属性设置,共提供有 16 种光标,见表 5-4。

表 5-4　cursor 属性

属　性　值	光　　标	属　性　值	光　　标
auto	默认值	se-resize	
default		ne-resize	
move		sw-resize	
pointer		nw-resize	
crosshair		s-resize	
e-resize		text	
n-resize		wait	
w-resize		help	

光标属性的应用示例为<style=cursor=″nw-resize″>、<style=″cursor=wait″>等。

5.3　应用 CSS 样式的 4 种方式

可以有 4 种方法将样式表的功能加到 Web 页面中。

5.3.1　直接定义 HTML 标记中的 style 属性

语法:

```
<HTML 标记名称 style=″样式属性 1:属性值 1;样式属性 2:属性值 2;…″>
```

5.3.2　在 HTML 文档内定义内部样式表

语法:

```
<style type=″text/css″>
<! --
    选择符 A1,选择符 A2,…{样式属性 1:属性值 1;样式属性 2:属性值 2;…}
    选择符 B1,选择符 B2,…{样式属性 1:属性值 1;样式属性 2:属性值 2;…}
    …
-->
</style>
```

CSS 选择符有 3 种:HTML 标记名称、class 选择符和 id 选择符。它们的定义与使用见表 5-5。

表 5-5　选择符的定义与使用

类　型	语　　法	说　　明	示　　例
HTML 标记	定义：标记{…} HTML 文件：<标记>	在 HTML 文件中，所有该标记处文本都具有定义的 CSS 样式	h3{color:red} …
class	定义：*.类名{…}或 .类名{…} HTML 文件： <标记 class=类名>	在 HTML 文件中，所有该类名处文本都具有定义的 CSS 样式	.am{color:red} <h3 class=am>…</h3>
	定义：标记.类名{…} HTML 文件： <标记 class=类名>	在 HTML 文件中，所有该标记并该类名处文本都具有定义的 CSS 样式	h3.am{color:red} <h3 class=am>…</h3>
id	定义：#标识{…} HTML 文件： <标记 id=标识>	在 HTML 文件中，所有该标识处文本都具有定义的 CSS 样式	#am{color:red} <h3 id=am>…</h3>
	定义：标记#类名{…} HTML 文件： <标记 id=类名>	在 HTML 文件中，所有该标识并该标记处文本都具有定义的 CSS 样式	h3#am{color:red} <h3 id=am>…</h3>

例 5.1　本例说明选择符的应用，ex5-01.html 代码清单如下：

```
<html><head>
  <title>选择符的应用</title>
</head>
<style type="text/css">   /*定义样式*/
<!--
   h2{ color:green;
       font-family:楷体;
     }
   .redfont{font-family:华文彩云;color:red}
   h4.bluefont{font-family:隶书;color:blue}
   #id_olivefont{font-family:楷体;color:olive}
   h4#purplefont{font-family:仿宋体;color:purple}
-->
</style>
<body>
<h2>显示楷体绿色</h2>
<h3 class=redfont>显示华文彩云红色</h3>
<h4 class=bluefont>显示隶书蓝色</h4>
<h3 id=id_olivefont>显示楷体橄榄绿</h3>
<h4 id=purplefont>显示仿宋体紫色</h4>
</body></html>
```

文件 ex5-01.html 在浏览器中的显示效果如图 5-2 所示。

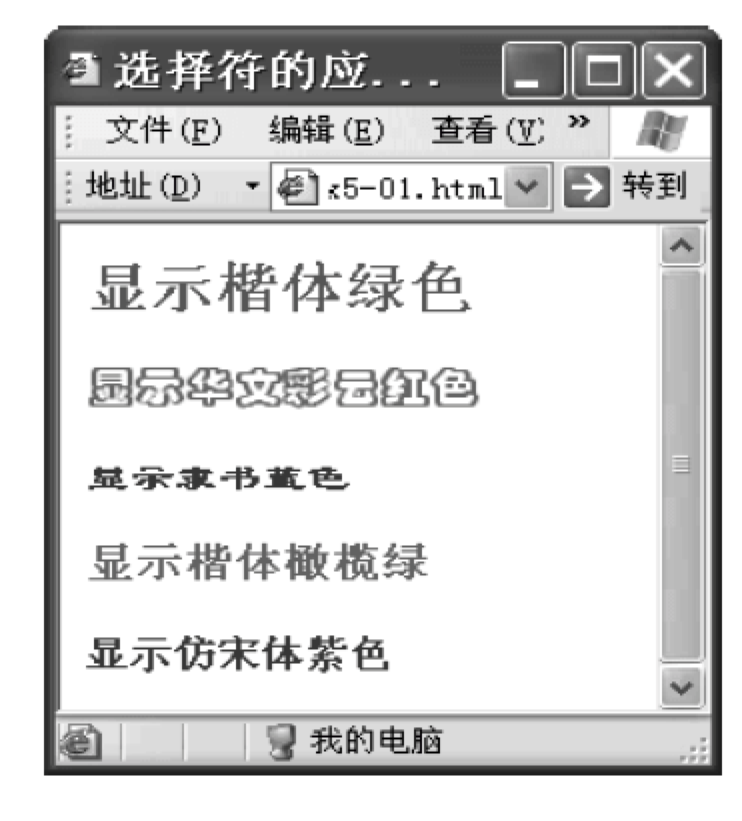


图 5-2 选择符的应用

5.3.3 嵌入外部样式表

语法：

```
<style type="text/css">
  <! --
    @import url("外部样式表文件名");
  -->
</style>
```

5.3.4 链接外部样式表

语法：

```
<link type="text/css" rel=stylesheet href="外部样式文件名">
```

5.4 样式表应用案例

使用直接定义 HTML 标记中的 style 属性、定义内部样式表、嵌入外部样式表和链接外部样式表这 4 种方法，实现如图 5-3 所示的功能。

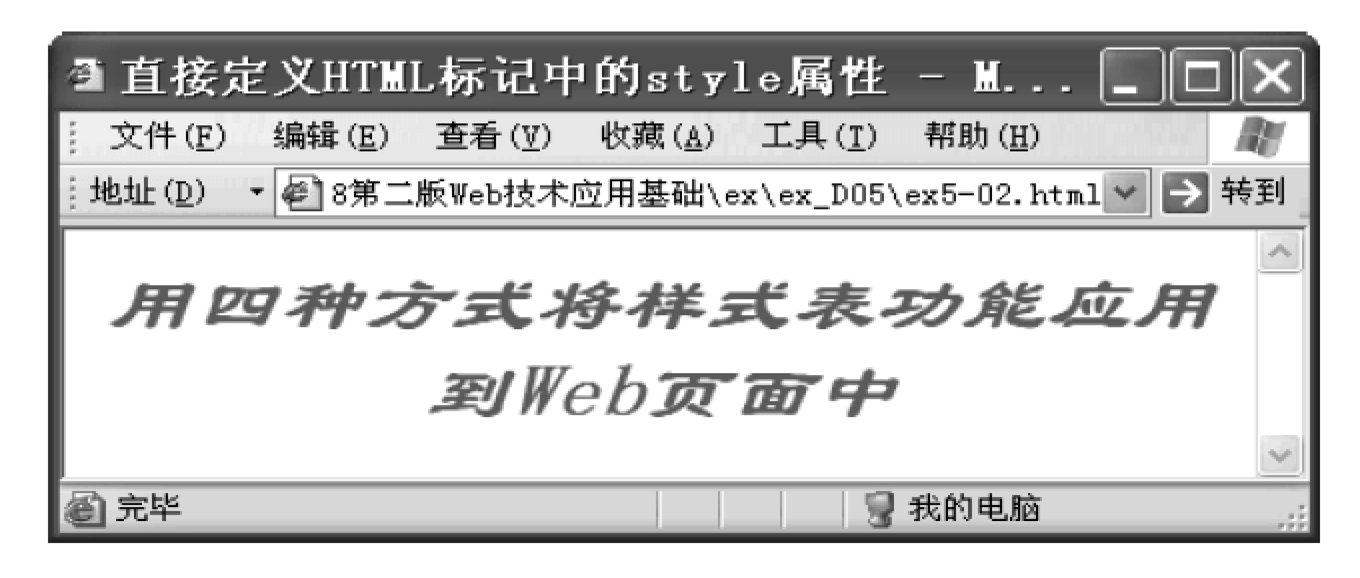


图 5-3 需要完成的 Web 页面

1. 直接定义 HTML 标记中的 style 属性

例 5.2 直接在 HTML 标记中插入 style 属性，只能控制该处的样式。ex5-02.html 代码清单如下：

```
<html><head>
  <title>直接定义 HTML 标记中的 style 属性</title>
</head>
<body>
```

```
<h1 style="color:green;text-align:center;font-style:italic;font-family:隶书;font-size:x-large;">
  用四种方式将样式表功能应用到 Web 页面中</h1>
</body></html>
```

2. 定义内部样式表

例 5.3 本例说明定义内部样式表的应用，ex5-03.html 代码清单如下：

```
<html><head>
  <title>定义内部样式表</title>
<style type="text/css">   /* 定义样式 */
<!--
   h1{color:green;
      text-align:center;
      font-style:italic;
      font-family:隶书;
      font-size:x-large;
     }
-->
</style>
</head>
<body>
  <h1>用四种方式将样式表功能应用到 Web 页面中</h1>
</body></html>
```

语法说明：

- 样式定义在文件头中，其有效范围是整个网页，且将样式设定与网页内容分开。
- 第 3 行<style type="text/css">，<style>标记的属性 type 指明使用 CSS。
- 在<style>标记内加上注释符号<!--……-->，可以使不支持 CSS 的浏览器忽略这段内容。

3. 嵌入外部样式表

当多个网页具有相同样式时，可以使用样式文件把设定的样式集中起来，并存成独立的样式文件，以使多个网页共享该样式文件。也可以将样式分类，使一个网页套用多个样式文件。

例 5.4 本例将建立两个样式文件，style1.css 文件保存文字的颜色，style2.css 文件保存文字的其他样式。

样式文件的后缀是.css，样式文件 style1.css 的代码清单如下：

```
h1  {color:green;
    }
```

样式文件 style2.css 的代码清单如下：

```
h1  {text-align:center;
```

```
        font-style:italic;
        font-family:隶书;
        font-size:x-large;
    }
```

文件 ex5-04.html 的代码清单如下:

```
<html><head>
<title>嵌入外部样式表</title>
<style type="text/css">   /*定义样式*/
<!--
   @import url("style1.css");
   @import url("style2.css");
-->
</style>
</head>
<body>
    <h1>用四种方式将样式表功能应用到 Web 页面中</h1>
</body></html>
```

4. 链接外部样式表

例 5.5 链接外部样式文件的 HTML 文件的页面代码 ex5-05.html 如下:

```
<html><head>
    <title>链接外部样式表</title>
    <link rel=stylesheet type="text/css" href="style1.css">
    <link rel=stylesheet type="text/css" href="style2.css">
</head>
<body>
    <h1>用四种方式将样式表功能应用到 Web 页面中</h1>
</body></html>
```

5.5 页 面 定 位

页面定位是指网页作者可以精确地将一个网页元素定位在页面的某一个位置。对 CSS 中几个页面定位元素说明如下。

(1) position:可以把页面元素精确地定位,它有 3 种设定方式 absolute/relative/static(绝对位置/相对位置/静态位置)。

- absolute:页面对象的绝对位置,原点在窗口的左上角。
- relative:页面对象与 HTML 代码中上一个对象的相对位置。
- static:静态位置,该值是默认值,页面对象将根据 HTML 源代码的位置顺序显示。

(2) left:元素的左边距。

(3) top：元素的顶边距。

(4) width：元素的宽度。

(5) height：元素的高度。

(6) z-index：定义页面元素在 Z 轴上的位置，Z 轴的定义从后到前。

例 5.6 制作一个页面，在页面中使文字有部分重叠。ex5-06.html 的代码清单如下：

```
<html><head>
<title>定位技术的应用</title>
<style type="text/css">
span{font-size:28pt;font-family:"隶书"}
span.level2{position:absolute;z-index:2;left:100;top:100;color:green}
span.level1{position:absolute;z-index:1;left:103;top:103;color:red}
span.level0{position:absolute;z-index:0;left:106;top:106;color:yellow}
p.lev1{position:absolute;z-index:2;left:50;top:50;font-size:20pt;color:blue}
p.lev2{position:absolute;z-index:-2;left:52;top:52;font-size:20pt;color:darkred}
</style>
</head>
<body>
<span class="level2">Web 技术应用基础</span>
<span class="level1">Web 技术应用基础</span>
<span class="level0">Web 技术应用基础</span>
<p class="lev1">欢迎学习</span>
<p class="lev2">欢迎学习</span>
</body></html>
```

在浏览器中的显示效果如图 5-4 所示。



图 5-4 网页元素定位

5.6 CSS 在网上书店案例中的应用

网上书店的各个页面采用了统一的风格。

(1) 超链接：设置没有访问过的超链接(a:link)的颜色是#FF9900，访问过的超链接

(a:visited)的颜色是＃0000FF，鼠标移到超链接(a:hover)上时的颜色是＃FF3300。

(2)．menufontcolor1：表示菜单栏字体颜色。

(3)．forminput：表示表单输入组件的大小。

(4)．formtext：表示表单文本框的字体。

(5)．td：表示单元格大小字体等。

样式文件 maincss.css 代码清单如下：

```
a:active {   text-decoration: none; color: #333333}
a:hover {   text-decoration: underline; color: #FF3300}
a:visited {   text-decoration: none; color: #0000FF}
a:link {
 color: #FF9900;
 text-decoration: none;
}
.menufontcolor1 {
 color: #333333;
}
.menufontcolor2 {
 color: #333333;
}
.forminput {
 height: 18px;
 border: 1px dotted;
}
.formtext {
 border: 1px dotted;
}
.td {
 border: 1px dotted;
 font-family: "宋体";
 font-size: 12px;
 color: #333333;
 text-decoration: none;
}
```

习题、上机练习 5

一、习题

1. 简述 CSS 的概念和它的功能。

2. 有如下一段代码，请问页面上的文字"Web 技术应用基础"显示成什么颜色？

```
<html><head></head>
```

```
<style type="text/css">
<! --
   font { color:red;
          font-family:楷体;
        }
--></style>
<body>
  <font>Web 技术应用基础</font>
</body></html>
```

3. 有哪几种方式可以把样式表的功能应用到页面中？

二、上机练习

1. 使用 CSS 技术制作一个页面，页面内容及显示格式如图 5-5 所示。

DHTML 技术(红色)

CSS 用于定义 Web 页面内容在浏览器上的显示方式，由于 HTML 的功能有限，一般不能随意设计版面和编排文字，所以 W3C 公布了一套 HTML 的扩展标准 CSS，扩展了 HTML 在排版和文字样式上的功能(蓝色)。

通过样式定义可以设定很多属性，如字号、颜色、页边距、元素在页面上的绝对位置等(绿色)。

图 5-5　页面内容

(1) 直接定义 HTML 标记中的 style 属性实现。

(2) 定义内部样式表实现。

(3) 嵌入外部样式表实现。

(4) 链接外部样式表实现。

2. 制作一个具有 3 个页面的网站，内容自定，应用 CSS 技术使它们具有相同的风格。

3. 制作一个页面，使图像的说明文字在图像之中，请注意文字颜色的选择，以便把图像和文字区分开。

第 6 章 JavaScript

脚本(Script)是一种介于 HTML 与编程语言(如 Java、VB 和 C++ 等)之间的特殊语言。尽管脚本更接近高级语言,但是它不具有高级语言复杂、严谨的语法规则,因而脚本语言易学易用,普及了 Web 应用的开发工作。脚本可以使 Web 页面具有动态效果和交互功能,分为服务器端脚本和客户端脚本两种。客户端脚本是在客户机上运行的脚本程序,在客户端产生动态效果;服务器端脚本是在服务器上运行的脚本程序,在客户端与服务器端交互产生动态效果。Script 语言一般指 VBScript 和 JavaScript。本章主要学习 JavaScript 的基础知识、工作机制及其使用技术。

6.1 JavaScript 概述

6.1.1 JavaScript 运行机制

JavaScript 和 Java 有一些相同之处,例如,它们的语法元素与 C++ 十分相似,都与平台无关。但是它们相异的地方也很多,例如,Java 是 Sun 的产品,而 JavaScript 是 Netscape 的产品;JavaScript 是解释型的,而 Java 是半编译半解释型的;从语法风格上看,JavaScript 比较自由,而 Java 是一种强类型语言,语法比较严谨。

JavaScript 是一种嵌入 HTML 文档中的、跨平台、基于对象和事件驱动的脚本语言,它既可以在客户端运行也可以在服务器上运行。在客户端执行脚本,需要在客户机上有使用脚本的应用程序和脚本引擎。使用脚本的应用程序(Script Host,也称为脚本主机)通常就是浏览器,它可以是 IE(Internet Explorer)或 IN(Netscape Navigator)。脚本引擎(Script Engine)通常是一个库或库的集合,例如,VBScript.DLL 或 JScript.DLL,由它们完成解释脚本的具体任务。脚本可被嵌入 HTML 文档,嵌入了脚本的 HTML 文档加载到浏览器内的解释器上,浏览器将把脚本程序交给脚本引擎执行。

JavaScript 由 JavaScript 核心语言、JavaScript 客户端扩展和 JavaScript 服务器端扩展三部分组成。核心语言部分包括 JavaScript 的基本语法和 JavaScript 的内置对象,在客户端和服务器端均可运行。客户端扩展部分是在 JavaScript 核心语言基础上扩展了控制浏览器的对象模型 DOM,在客户端运行脚本时,可以很方便地控制页面上的对象。服

务器端扩展部分是在 JavaScript 核心语言基础上扩展了在服务器上运行时需要的对象，这些对象可以和 Web 数据库连接，可以在应用程序之间交换信息，对服务器上的文件进行控制。

6.1.2 JavaScript 的特点

JavaScript 具有以下特点。

1. JavaScript 是一种脚本语言

JavaScript 是一种脚本语言，脚本是一种能够完成某些功能的小程序段。脚本语言可嵌入 HTML 页面，这些程序段在程序运行过程中被逐行解释执行。在脚本中使用的命令和语句集被称为脚本语言。

2. 基于对象

JavaScript 是基于对象的(Object-Based)，允许自定义对象。同时浏览器也提供了大量的内建对象，开发人员可以直接使用，例如，string(字符串)、Date(日期)和 math(算术)对象等。但 JavaScript 不是面向对象(Object-Oriented)的，它不支持类和继承。

3. 事件驱动

事件是用户在操作 Web 页面时发生的任何事情，例如，鼠标左键单击、鼠标拖动、窗口移动或表单输入等。JavaScript 是事件驱动的，当事件发生时，即可对事件产生反应并进行处理。

4. 动态

JavaScript 是动态的，可以直接对用户的输入做出响应，是设计交互式动态尤其是“客户端动态”页面的重要工具。

5. 安全

JavaScript 是一种安全性的语言，它不允许访问本地硬盘，不能修改或删除其他文件的内容，不能将数据存储在 Web 服务器或用户的计算机上。

6. 与平台无关

JavaScript 只依赖于浏览器，与操作环境无关。只要是能运行支持 JavaScript 的浏览器的计算机，就可以正常工作。

6.1.3 JavaScript 应用案例——图像互换位置

例 6.1 应用 JavaScript 制作第一个脚本程序。

1．任务要求

要求页面上有两幅图像及有关图像的说明的文字，当用户单击页面时，图像交换位置。页面如图 6-1 所示。

分析任务要求，确定本例需要使用的技术。由于页面上有两幅图像，所以需要使用块容器标记<div>界定这两幅图像和它的文字说明，并使用绝对定位技术将图像定位。鼠标单击页面时，产生鼠标事件，图像互换位置的动作是对鼠标事件的响应，本例将使用 JavaScript 编制事件处理程序。



图 6-1 JavaScript 应用案例的页面

2．完成以上任务的代码 ex6-01.html

代码 ex6-01.html 清单如下：

```
<html><head>
<title> JavaScript 应用案例</title>
<script language="JavaScript">
function ChangeImage(){
    var dog_top=dog.style.top
    var dog_left=dog.style.left
    dog.style.top=cat.style.top
    dog.style.left=cat.style.left
    cat.style.top=dog_top
    cat.style.left=dog_left
}
</script>
</head>
<body onclick="ChangeImage()"> 
<font face="隶书" color="blue" size=6>请单击页面</font><p>
<div id="cat" style="position:absolute;top:60px;left:60px">
    <img src="cat.gif" width="80"  height="80" > <br>
    <font size=5 color="red">cat</font>
</div>
<div id="dog" style="position:absolute;top:60px;left:160px">
    <img src="dog.gif" width="80"  height="80"><br>
    <font size=5 color="red">dog</font>
</div>
</body></html>
```

3．代码说明

(1) 第 16 行至第 19 行：

<div>和</div>是块容器标记，之间可以容纳多个不同的 HTML 标记和语言

元素。

id="cat",指定该块容器的 id 值是 cat。

(2) 第 3 行至第 12 行:当用户单击页面时,由使用 JavaScript 脚本语言编写的 ChangeImage()函数完成事件的处理,即图像交换位置的动作。

6.2 JavaScript 基本语法

6.2.1 在 HTML 文档中调入或嵌入 JavaScript

JavaScript 是通过嵌入或调入在标准的 HTML 语言中的可执行程序段。

1. JavaScript 嵌入 HTML 文件

应用 HTML 的<script>标记嵌入 JavaScript 代码。

语法:

```
<script language="JavaScript">
  JavaScript 代码
</script>
```

将 JavaScript 代码嵌入 HTML 文档,使用 HTML 的<script>标记的 language 属性,属性值可以是 JavaScript,也可以是 VBScript。<script>可以包含在<head>和<body>内。包含在<head>内的 JavaScript 脚本在页面装载之前运行,所以函数一般包含在<head>标记之间。

2. 将 JavaScript 调入 HTML 文件

将 JavaScript 代码以扩展名".js"单独存放,再用<script>标记的 src 属性把该代码调入 HTML 文档。

语法:

```
<script src="JavaScript 文件名">
```

例 6.2 将 JavaScript 嵌入 HTML 文档,ex6-02.html 代码清单如下:

```
<html><head>
<title> JavaScript 嵌入 HTML</title>
</head>
<body>
 <script language="JavaScript">
    document.write("Hello World!")
 </script>
</body></html>
```

例 6.2 的第 6 行"document.write("Hello World!")"应用了 JavaScript 的 document 对象的 write()方法,把字符串"Hello World!"输出到浏览器窗口中显示。例 6.2 运行效果如图 6-2 所示。

6.2.2 JavaScript 书写格式

(1) JavaScript 是区分大小写的。

(2) JavaScript 没有可见的行结束标志,它用换行符作为一行的终止符。

图 6-2 将 JavaScript 嵌入 HTML

(3) 如果需要把几行代码写在一行中,使用分号(;)把它们分开。例如:

```
var a=3
var b=6
var c=0
```

可以写成

```
var a=3; b=6;c=0
```

它们的结果是一样的。

(4) 为了使程序清晰易读,采用缩进格式来书写。

(5) 可以用两种方法进行注释。注释方法与 C++ 相同。

//:从注释标记"//"起直到行尾的字符在脚本运行时都被忽略。

/ * … * /:在"/ * "与" * /"之间的字符在脚本运行时都被忽略。

6.2.3 基本数据类型

JavaScript 和其他语言一样,有自己的基本数据类型、表达式、算术运算符以及程序基本框架结构。

1. 数据类型

JavaScript 有 4 种基本数据类型:数值型(整数和浮点数)、字符型(用" "或' '括起来的字符或数值)、布尔型(取值为 true 或 false)和空值。基本类型中的数据可以是常量,也可以是变量。JavaScript 采用弱类型(Loosely typed)形式,允许一个变量或常量在使用时确定它的数据类型。

- 数值型:数值型包括整数和浮点数。
  整数可以是十进制、八进制和十六进制数,八进制数以 0 开头,十六进制数以 0x 或 0X 开头。例如:189(十进制),056(八进制),0x9f(十六进制)。
  浮点数用来表示小数、极大或极小的数,例如:−3.1416,66.,1.6e3,7e−9 等。
- 字符型:字符型数据的值是用(" ")或(' ')括起来的一连串字符或数字,例如:'c','d',"123456","Hello World!"等。其中"\"是转义字符,例如,'\n'是换行符,

'\t'是水平制表符,'\r'是回车符,'\b'是退格符等。

- 布尔型：布尔型的数据有 true 和 false 两种,分别表示逻辑真和逻辑假。
- 空值：空值 null 表示什么也没有。如果企图引用一个没有定义的变量,就会返回一个 null 值。

2. 常量

JavaScript 常量又称字面量,是其值保持不变的量。

3. 变量

变量是在代码中值可以改变的量。JavaScript 中使用关键字 var 声明变量并分配存储空间。变量名必须以字母或下划线"_"开始,后面的字符可以是字母、数字或下划线。JavaScript 内部定义的保留字不能用作变量名,JavaScript 的保留字如下：

abstract	boolean	break	byte	case
catch	char	class	const	continue
default	do	double	else	extends
false	final	finally	float	for
function	goto	if	implements	import
in	instanceof	int	interface	long
native	new	null	package	private
protected	public	return	short	static
super	switch	synchronized	this	throw
throws	transient	true	try	var
void	while	with		

JavaScript 是区分大小写的,如变量 Num 与变量 num 是两个不同的变量。

语法：

```
var  变量名
var  变量名=变量值
```

例如：

```
var a=3
str="Hello World!"
num1=12
num2=3.14
var d,b,c=0
```

在 JavaScript 中变量声明关键字是可选的,例如语句 var a=3 也可以写成 a=3。但是不能既不用关键字 var 声明也不给它赋初值。

JavaScript 的变量采用了弱类型表达方式,变量在声明时不必显式说明它的类型,而是在使用时根据数据的类型来确定变量的类型。

在 JavaScript 中既可以在声明变量时初始化，也可以在变量被声明后赋值。例如，语句：

```
var num1=6
```

和语句：

```
var num1
num1=6
```

的作用是一样的。

4. 表达式

表达式是由变量、常量、布尔和运算符按一定规则组成的集合，表达式的值可以是数字、字符串或布尔量。JavaScript 有 3 种表达式：算术表达式、串表达式和逻辑表达式。例如：

```
num1+23
"Hello"+"World!"
A && B
```

5. 运算符

JavaScript 的运算符有：赋值运算符、算术运算符、逻辑运算符、比较运算符、字符串运算符和位运算符。表 6-1 列出了 6 种赋值运算符的使用说明。

表 6-1　赋值运算符的使用说明

运　算　符	范　　例	使 用 说 明
=	z=x+y	把=号右边表达式的值赋给左边的变量
+=	x+=y	x+=y 与 x=x+y 等同
-=	x-=y	x-=y 与 x=x-y 等同
*=	x*=y	x*=y 与 x=x*y 等同
/=	x/=y	x/=y 与 x=x/y 等同
%=	x%=y	x%=y 与 x=x%y 等同(x 除以 y，把余数赋给 x)

表 6-2 列出算术运算符的使用说明。

表 6-2　算术运算符的使用说明

运　算　符	范　　例	使 用 说 明
+	1+2，-23.6+678	加
-	654-234，345.6-56.8	减
*	34*56，45.6*56.7	乘

续表

运　算　符	范　　例	使 用 说 明
/	64/8,456.9/34.5	除
%	5%3	求模(求除法的余数)
＋＋	x＋＋,＋＋x	一元自加,把操作数加 1
－－	x－－,－－x	一元自减,把操作数减 1

表 6-3 列出了逻辑运算符的使用说明。

表 6-3　逻辑运算符的使用说明

运　算　符	范　　例	使 用 说 明
&&	A&&B	逻辑与,当操作符两边表达式的值均为 true 时,结果为 true,否则结果为 false
‖	A ‖ B	逻辑或,A 和 B 中有一个的值为 true 时,结果为 true;A 和 B 都是 false,结果为 false
!	! A	求反,若 A 的值为 true,结果为 false;A 的值为 false,结果是 true

表 6-4 列出了比较运算符的使用说明。

表 6-4　比较运算符的使用说明

运　算　符	范　　例	使 用 说 明
＝＝	A＝＝B	相等比较,如果变量 A 和 B 相等,结果为 true;否则,结果为 false
!＝	A!＝B	不相等比较,如果变量 A 和 B 不相等,结果为 true;否则,结果为 false
＞	A＞B	大于比较,如果变量 A 大于 B,结果为 true;否则,结果为 false
＜	A＜B	小于比较,如果变量 A 小于 B,结果为 true;否则,结果为 false
＞＝	A＞＝B	大于等于比较,如果变量 A 大于等于 B,结果为 true;否则,结果为 false
＜＝	A＜＝B	小于等于比较,如果变量 A 小于等于 B,结果为 true;否则,结果为 false

JavaScript 只有一个字符串运算符“＋”,使用字符串运算符可以把几个串连接在一起。例如,“字符串”＋“运算符”的返回值是“字符串运算符”。

表 6-5 列出了位运算符的使用说明。

表 6-5　位运算符的使用说明

<table>
<tr><th>运　算　符</th><th>范　例</th><th>使 用 说 明</th><th>运　算　符</th><th>范　例</th><th>使 用 说 明</th></tr>
<tr><td>&</td><td>A&B</td><td>按位与运算</td><td>～</td><td>～A</td><td>按位取反运算</td></tr>
<tr><td>|</td><td>A|B</td><td>按位或运算</td><td>＜＜</td><td>A＜＜B</td><td>左移运算</td></tr>
<tr><td>^</td><td>A^B</td><td>按位异或运算</td><td>＞＞</td><td>A＞＞B</td><td>右移运算</td></tr>
</table>

JavaScript 中,运算符优先级从高到低如下排列:

- 对象访问：.
- 调用和成员：()　[]
- 求反和增量：!　~　-　++　--
- 乘、除和取模：*　/　%
- 加和减：+　-
- 位移位运算：>>　<<
- 比较运算：<　<=　>　>=
- 等于和不等于：==　!=
- 按位与：&
- 按位异或：^
- 按位或：|
- 逻辑与：&&
- 逻辑或：||
- 条件运算：?　:
- 赋值：=　+=　-=　*=　/=　%=　<<=　>>=　&=　^=　|=
- 逗号：,

6.3　JavaScript 控制结构和函数

6.3.1　JavaScript 控制结构

JavaScript 有 4 种程序结构控制语句：if、switch、for 和 while。

1. 条件语句 if…else

语法如下：

```
if(条件){
    //如果条件为 true,进行处理
}
else{
    //如果条件为 false,进行处理
}
```

如果不需要处理条件为 false 的情况，可以不写 else 语句段，例如：

```
if(条件){
    //如果条件为 true,进行处理
}
```

if 语句可以嵌套。

2. switch 语句

switch 语句用于经过测试后决定哪条语句被执行。语法如下：

```
switch(表达式){
    case 值 1:
        语句集 1
        break
    case 值 2:
        语句集 2
        break
    …
    default:
        语句集
        break
}
```

3. 循环语句 for

for 循环语句设置了一个计数器计算循环次数，达到循环次数后结束循环。它的语法规则如下：

```
for(初始化表达式;条件;增量表达式){
      语句集
}
```

4. 循环语句 while

while 循环语句不直接指明循环次数，具体循环次数由运行时情况决定，满足循环条件执行循环体语句，不满足循环条件退出循环体。它的语法规则如下：

```
while(条件){
    语句段
}
```

- break 语句：根据条件终止循环。
- continue 语句：根据条件，跳过循环体内剩余语句，进入下一次循环。

6.3.2 函数

在 JavaScript 中可以使用函数，函数是封装在程序中可以多次使用的模块。函数必须先定义后使用。由于浏览器先执行 HTML 文档中的<head>模块，所以 JavaScript 中使用函数时，常把自定义函数放在<head>模块中，然后在 HTML 文档的主体<body>模块中调用函数。

函数定义的规则如下：

```
function 函数名(参数列表){
    函数体
}
```

- function：是 JavaScript 的关键字，定义一个函数。
- 函数名：函数名跟在关键字 function 后面，它可以是任何合法的标识符。
- 参数列表：函数的参数列表，多个参数用逗号分开。
- 函数体：该函数执行的运算。

6.3.3 JavaScript 基本语法应用案例

例 6.3 说明 JavaScript 基本语法的应用方法，包括程序结构和函数，ex6-03.html 代码清单如下：

```
<html>
<head><title>JavaScript 基本语法应用案例</title>
<script language="JavaScript">              //脚本语言是 JavaScript
function MyArray(n){                        //定义函数 MyArray
  this.length=n
  for(i in 4)
    this[i]=0
}
MyArray=new Array(4)
MyArray[1]="Web"
MyArray[2]="技术"
MyArray[3]="应用"
MyArray[4]="基础!"
document.open()
for(var n=1;n<MyArray.length;n++){
  document.write(MyArray[n]);
}
document.close()
</script></head>
<body></body></html>
```

ex6-03.html 代码在浏览器中的显示效果如图 6-3 所示。



图 6-3　JavaScript 基本语法应用案例

6.4 JavaScript 对象

Java 是面向对象的，而 JavaScript 是基于对象的，所以 JavaScript 没有提供抽象、继承和重载等面向对象语言的功能。在 JavaScript 中，对象是客观事物的描述，它有内建对

象和用户自定义对象两大类。

6.4.1 JavaScript 对象概述

JavaScript 对象是对具有相同特性的实体的抽象描述，对象实例是具有这些特征的单个实体。对象包含属性(properties)和方法(methods)两种成分。属性是对象静态特性的描述，是对象的数据，以变量表征；方法是对象动态特性的描述，也可以是对数据的操作，用函数描述。

对象必须存在，才能够被引用，有以下 3 种方法引用对象：

- 引用 JavaScript 内建对象。
- 引用浏览器环境提供的对象。
- 创建自定义对象。

6.4.2 自定义对象

JavaScript 可以根据需要创建自定义对象。创建方法是：先定义一个对象，然后创建该对象的实例。

1. 定义对象

在 JavaScript 中应用 function 关键字创建用户自定义对象。

语法：

```
function 对象名称(属性列表){
    this.属性 1=参数 1
    this.属性 2=参数 2
    …
    this.方法 1=函数名 1
    this.方法 2=函数名 2
    …
}
```

例如，学生对象的定义：

```
function student(id,name,url){
    this.id=id
    this.name=name
    this.url=url
    this.display=student_display
}
```

2. 创建对象实例

对象定义后应用关键字 new 创建对象实例。

语法：

对象实例名＝new 对象名称(属性值列表)

例如：

```
MyStudent＝new student("000001","林琳","http://www.buu.com.cn")
```

创建了一个 student 对象的 MyStudent 实例。

6.4.3 对象属性和方法的引用

对象实例创建后，可以通过该实例引用对象的属性和方法。

1. 对象属性的引用

对象属性的引用可以有两种方式。

(1) 使用(.)运算符

语法：

对象实例名.属性成员名

例如：

```
MyStudent.name＝"林琳"
```

(2) 通过对象实例的下标引用

语法：

对象实例名[n]

例如：

```
MyStudent[0]＝ "000001"
MyStudent[1]＝ "林琳"
MyStudent[2]＝http://www.buu.com.cn
```

或：

```
MyStudent["id"]＝ "000001"
MyStudent["name"]＝ "林琳"
MyStudent["url"]＝http://www.buu.com.cn
```

2. 对象方法的引用

使用(.)运算符引用对象方法。

语法：

对象实例名.方法名称()

例如：

```
MyStudent.display()
```

6.4.4 对象的操作

在 JavaScript 中提供了操作对象的语句、关键字和运算符。

1. for…in 语句

JavaScript 是基于对象的语言，对象由与之相关的属性和方法组成。for…in 是操作对象的语句，也称遍历循环。遍历循环是指历经一个集合体中的每个个体。JavaScript 中的遍历是指逐一通过一个对象的所有属性，它的计数值是对象中的属性个数。for…in 语句的语法规则如下：

```
for(变量名 in 对象实例名){
    语句段
}
```

2. with 语句

如果在程序中需要连续使用某个对象的一些属性和方法，可以使用 with 语句。

语法如下：

```
with(对象实例名){
    语句段
}
```

例如：

```
with(MyStudent){
    id="000001"
    name="林琳"
    url=http://www.buu.com.cn
}
```

3. this 关键字

this 关键字是对当前对象的引用。

6.4.5 事件驱动与事件处理

JavaScript 是基于对象的，采用事件驱动(event driven)机制。事件是对计算机进行的操作，例如，鼠标移动、鼠标左键单击或热键动作等都是事件。由鼠标或热键引发的一

连串的动作称为事件驱动。处理事件的程序或函数称为事件处理程序(event handler)。

事件是用户与 Web 页面的交互操作,例如用户单击超链接或提交表单控件数据时,产生一个事件,并通知浏览器对该操作进行处理。浏览器等待事件发生,在事件发生时进行处理。JavaScript 常用事件见表 6-6。

表 6-6　JavaScript 常用事件

事　　件	说　　明
onClick	鼠标左键单击页面对象时发生
onload	网页载入浏览器时发生,发生对象为 HTML 的<body>标记
onUnload	用户离开当前页面时发生,发生对象为 HTML 的<body>标记
onMouseOver	鼠标移到对象上时发生
onMouseOut	鼠标离开对象上时发生

例如,例 6.1 中的<body onclick="ChangeImage()">语句,用鼠标左键单击页面事件,引发“ChangeImage()”事件处理程序,使图像互换位置。

6.4.6　JavaScript 对象应用案例

例 6.4　说明 JavaScript 对象的使用方法,ex6-04.html 代码清单如下:

```
<html>
<head><title>JavaScript 对象应用</title>
<script language="JavaScript">
   function student(id,name,url){
        this.id=id
        this.name=name
        this.url=url
        this.display=student_display
   }
   function student_display(){
        document.writeln("id="+this["id"]+"<br>")
        document.writeln("name="+this["name"]+"<br>")
        document.writeln("url="+this["url"]+"<br>")
   }
    MyStudent = new student("000001","林 琳","http://www.buu.com.cn")
   MyStudent.display()
</script></head>
<body></body></html>
```

ex6-04.html 代码在浏览器中的显示效果如图 6-4 所示。



图 6-4　JavaScript 对象应用

6.5 window 对象在 JavaScript 中的应用

6.5.1 window 对象的构成

对象有用户创建的对象,也有系统提供的内建对象。window 对象是内建对象中的最顶层对象,指的是浏览器窗口对象。它下层的对象有 event 对象、frame 对象、document 对象等,其中最主要的对象是 document 对象,它指的是 HTML 页面对象。window 和 document 对象的结构见图 6-5。

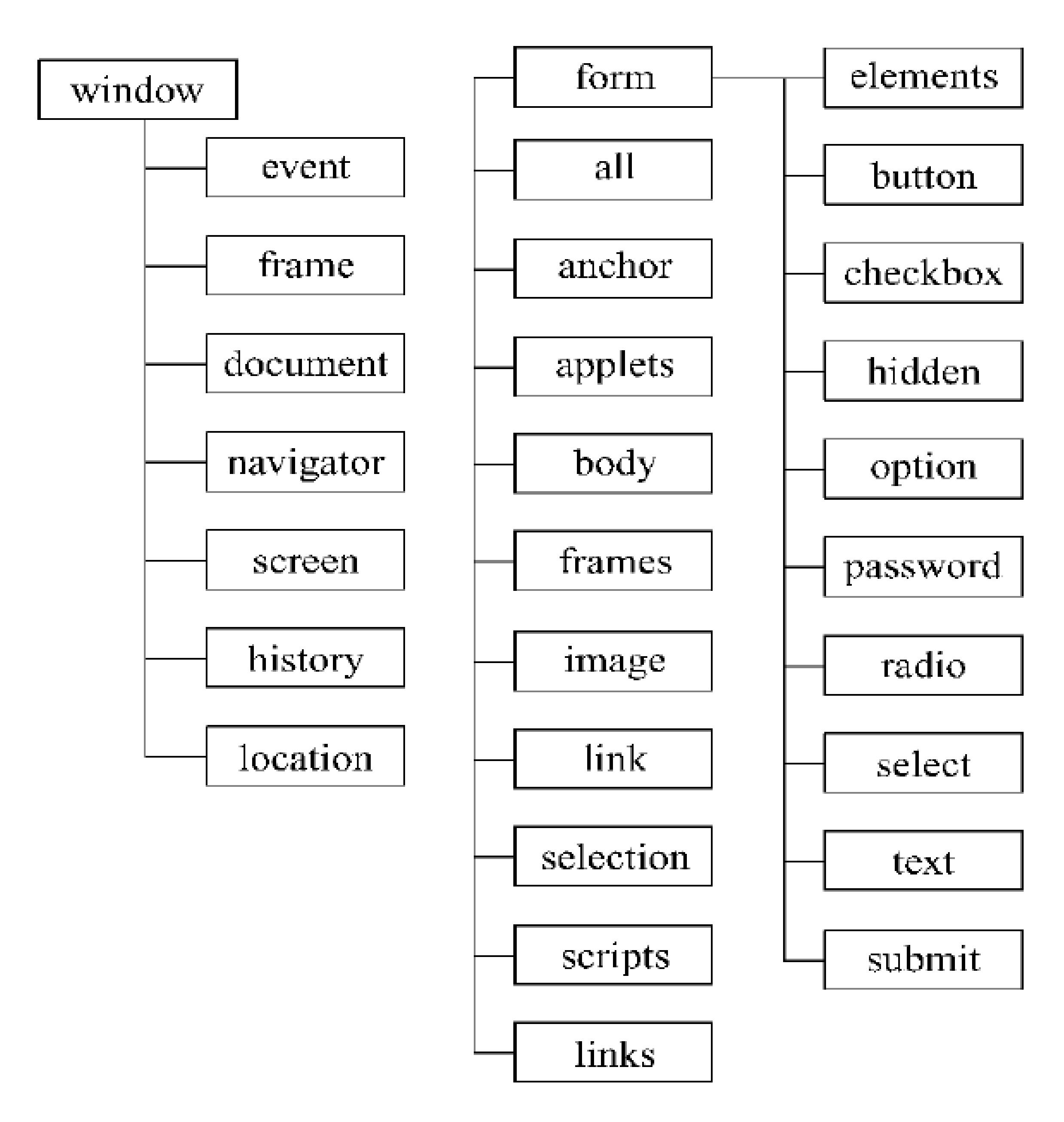


图 6-5 window 对象的结构

6.5.2 window 对象的属性

window 对象的属性主要有 name、parent、self、top、status 和 defaultStatus 等,它的方法主要有 alert()、confirm()、close()、prompt()、setTimeout()和 clearTimeout()等。

表 6-7 列举了 window 对象的主要属性和它们的应用说明。

表 6-7 window 对象的属性及其应用说明

属性名称	说明	范例
name	当前窗口的名字	window. name
parent	当前窗口的父窗口	parent. name

续表

属性名称	说　　明	范　　例
self	当前打开的窗口	self.status="你好"
top	窗口集合中的最顶层窗口	top.name
status	设置当前打开窗口状态栏的显示数据	self.status="欢迎"
defaultStatus	当前窗口状态栏的显示数据	self.defaultStatus="欢迎"

6.5.3　window 对象的方法

1. window 对象的主要方法

表 6-8 列举了 window 对象的主要方法和它们的应用说明。

表 6-8　window 对象方法及其应用说明

方法名称	说　　明	范　　例
alert()	创建一个带"确定"按钮的对话框	window.alert("输入错误!")
confirm()	创建一个带"确定"和"取消"按钮的对话框	window.confirm("是否继续")
close()	关闭当前打开的浏览器窗口	window.close()
open()	打开一个新的浏览器窗口	window.open(URL,"新窗口名",新窗口设置)
prompt()	创建一个带"确定"、"取消"按钮及输入字符串字段的对话框	window.prompt("请输入电话号码")
setTimeout()	设置一个时间控制器	window.setTimeout("clearTimeOut()",3000)
clearTimeout()	清除原来时间控制器内的时间设置	window.clearTimeout()

2. JavaScript 的接口元素

window 对象方法中的 alert()、prompt()和 confirm()方法,用作 JavaScript 的接口元素,用来显示用户的输入,并完成用户和程序的对话过程。

- alert(提示):显示一个警告框,其中"提示"是可选的,是在警告框中输出的内容。
- prompt(提示,默认值):显示一个提示框,等待输入文本,如果选择"确认"按钮,返回文本框中的内容,如果选择"取消"按钮,返回一个空字符串。它的"提示"和"默认值"都是可选的,"默认值"是文本框的默认值。
- confirm(提示):显示一个确认框,等待用户选择按钮。"提示"也是可选的,是在提示框中显示的内容,用户可以根据提示选择"确定"或"取消"按钮。

6.5.4 window 对象的事件

window 对象的主要事件及它们的应用见表 6-9。

表 6-9 window 对象的事件及其应用说明

事 件	应 用 说 明	事 件	应 用 说 明
onLoad	网页载入浏览器时发生	OnResize	用户调整窗口大小时发生
onUnLoad	网页从浏览器窗口中删除时发生	OnScroll	用户滚动窗口时发生
onBeforeUnLoad	网页被关闭前发生	OnError	载入的网页产生错误时发生

6.5.5 window 对象的应用案例

1. 状态栏内容的更新

例 6.5 要求页面上有一个按钮，把鼠标移动到按钮上时，状态栏输出“欢迎学习 Web 技术!”，2000ms 后状态栏输出“学习成功!”。

该任务使用了 window 对象的 status 属性，设置当前打开窗口状态栏的显示数据；window 对象的 setTimeOut()和 clearTimeOut()方法，设置一个时间控制器，控制状态栏内容的显示时间；onMouseOver 事件，当鼠标移动到按钮上时发生的事件。要求页面如图 6-6 所示。

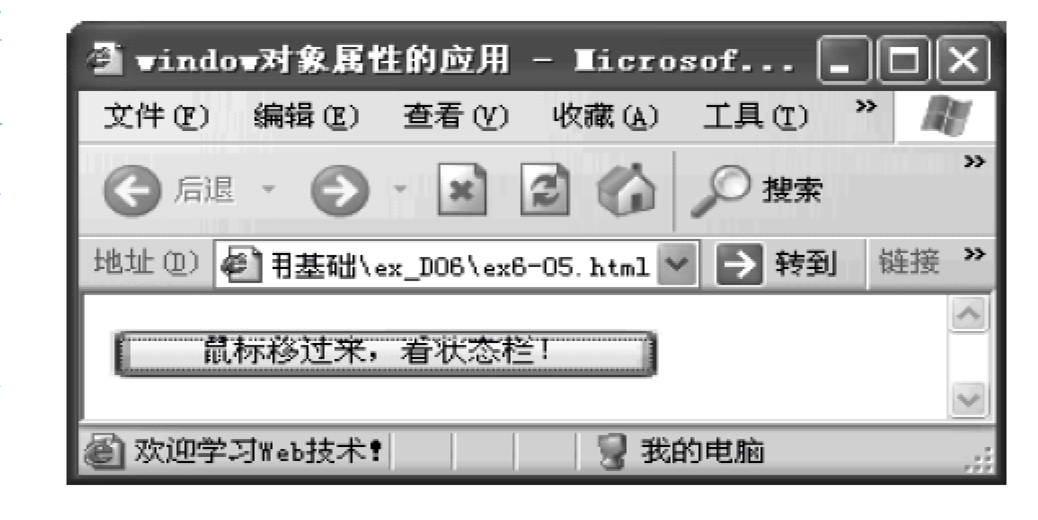


图 6-6 状态栏属性的应用

完成该任务的页面代码 ex6-05.html 清单如下：

```
<html><head>
<title>window 对象属性的应用</title>
<script language="JavaScript">
<! --
function clearStatus(){
    window.status="学习成功!"
}
function writeStatus(str){
    setTimeout("clearStatus()",2000)
    window.status=str
}
-->
</script></head>
<body>
```

```
<form>
<input type="button"  name="ControlButton" value="鼠标移过来,看状态栏!"
onMouseOver="writeStatus('欢迎学习 Web 技术!');return true;">
</form>
</body></html>
```

代码说明:

(1) 第 17 行:onMouseOver="writeStatus('欢迎学习 Web 技术!');return true;",onMouseOver 是当鼠标移动到按钮上时发生的鼠标事件,赋值号右边是对鼠标事件的响应,由文件头中的 JavaScript 程序完成。

(2) 第 5 行至第 11 行是用 JavaScript 编制的事件响应程序。window.status 是 window 对象的 status 属性,用于设置状态栏的输出。setTimeOut()和 clearTimeOut()是 window 对象的方法,用来设置和清除时间控制器。

2. 打开一个新窗口

例 6.6 要求页面上有一个按钮,把鼠标移动到按钮上时,打开一个新窗口。

本实例使用 window 对象的 open 方法,打开了 Tomcat 的欢迎页面。页面如图 6-7 所示。

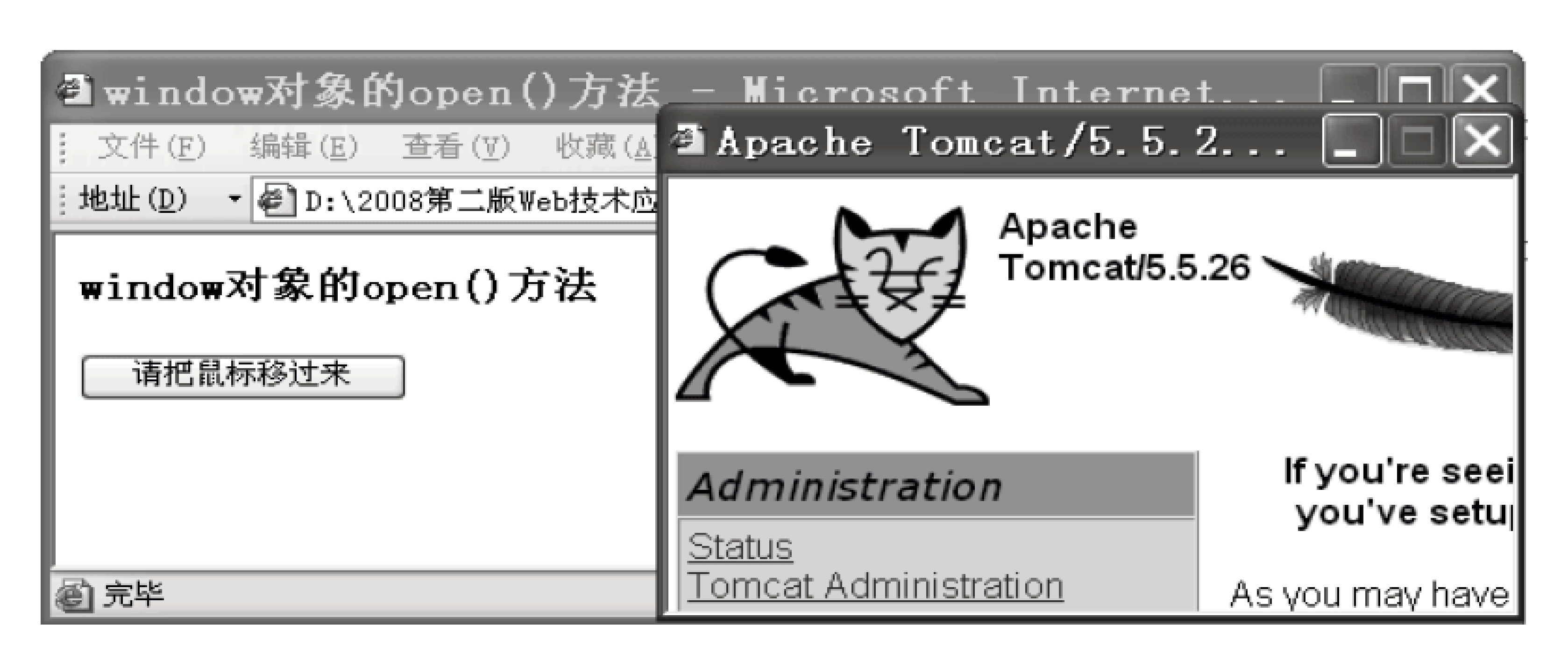


图 6-7 open 方法的应用

完成任务的页面代码 ex6-06.html 清单如下:

```
<html><head>
<title>window 对象的 open()方法</title>
</head>
<body>
<font size=4><b>
<p>window 对象的 open()方法</p>
<form><input type="button" name="ControlButton" value="请把鼠标移过来"
          onMouseOver="window.open('http://127.0.0.1:8080', '新窗口','width=350,
          height=200')">
</form>
</b></font>
</body></html>
```

代码说明：第 8 行的 http://127.0.0.1:8080 指明新窗口的 URL。

3. 客户端输入信息验证

例 6.7 在客户端验证用户输入数据。页面上有一个超级链接，单击链接时，由 prompt 提示框提示用户输入姓名，然后 JavaScript 程序验证用户输入，如果输入正确弹出确认框 confirm，若用户在确认框选择“确认”按钮，则链接到相关网站；如果输入错误，出现警告框 alert，输出“对不起，输入错误。”，程序终止。

通过本案例，将学习 JavaScript 接口元素 prompt、confirm 和 alert 的使用。

实现上述任务的代码 ex6-07.html 清单如下：

```
<html><head>
<title>JavaScript 接口元素应用</title>
<script language="JavaScript">
<! --
  function MyLink(MyName){
      var MyString=prompt("请输入姓名：")
      if(MyString==MyName)/* 验证输入姓名是否正确 */
      {    var Mybool=confirm(MyString +"你好！链接到 tomcat 页面?")
           if(! Mybool)
               window.event.returnValue=false
      }
      else
      {    alert("对不起，用户名错误。")
           window.event.returnValue=false
      }
  }
-->
</script>
</head>
<body>
<center><a href="http://127.0.0.1:8080" onClick=MyLink("林琳")>
<h2>你好!欢迎光临 tomcat! </h2></a>
</center>
</body></html>
```

代码说明如下：

- 程序中第 13 行的：alert("对不起，用户名错误。")，它在浏览器中显示如图 6-8 所示。
- prompt(提示，默认值)：第 6 行的 var MyString=prompt("请输入姓名：")，如果用户选择“确认”按钮，在提示框中输入的数据赋给变量 MyString，它在浏览器中显示如图 6-9 所示。



图 6-8 警告框



图 6-9　提示框

- confirm(提示)：第 8 行的 var Mybool＝confirm(MyString ＋"你好！链接到 tomcat 页面?")，该确认框在浏览器中的显示如图 6-10 所示。



图 6-10　确认框

- 第 14 行：window.event.returnValue＝false，如果不想建立链接，单击"取消"按钮，window.event.returnValue 将 false 传给浏览器，浏览器将忽略"按下链接"的动作，不产生建立链接的动作。

6.6　document 对象在 JavaScript 中的应用

window 对象的下层中使用最多的是 document 对象。

6.6.1　document 对象的属性

document 对象的属性可以用来设置 Web 页面的特性，例如标题、前景色、背景色和超链接颜色等。它主要用来设置当前下载的 HTML 文件中的基本数据和字符串的显示效果。表 6-10 列举了 document 对象的主要属性和它们的使用说明。

表 6-10　document 对象属性及其应用

属性名称	说　明	范　例
alinkColor	页面中活动超级链接的颜色	document.alinkColor＝"red"
bgColor	页面背景颜色	document.bgColor＝"ff0000"
fgColor	页面前景颜色	document.fgColor＝"ff000F"
linkColor	页面中未曾访问过的超级链接的颜色	document.linkColor＝"red"
vlinkColor	页面中曾经访问过的超级链接的颜色	document.vlinkColor＝"green"
lastModified	最后一次修改页面的时间	date＝lastModified
location	页面的 URL 地址	url_info＝document.location
title	页面的标题	tit_info＝document.title

6.6.2 document 对象的方法

表 6-11 列举了 document 对象的主要方法和它们的使用说明。

表 6-11 document 对象方法及其应用

方法名称	说　明	范　例
clear()	清除文件窗口内的数据	document.clear()
close()	关闭文档	document.close()
open()	打开文档	document.open()
write()	向当前文档写入数据	document.write("你好!")
writeln()	向当前文档写入数据,并换行	document.writeln("你好!")

6.6.3 document 对象的事件

表 6-12 列举了 document 对象的鼠标事件和它们的使用说明。

表 6-12 document 对象鼠标事件及其应用

鼠标事件	使用说明	鼠标事件	使用说明
onClick	在页面上单击鼠标左键时发生	onMouseOver	鼠标移到对象上时发生
ondblClick	在页面上双击鼠标左键时发生	onMouseUp	放开鼠标左键时发生
onMouseDown	在页面上按下鼠标左键时发生	onSelectStart	开始选取对象内容时发生
onMouseMove	在页面上移动鼠标时发生	onDragStart	以拖曳方式选取对象时发生
onMouseOut	鼠标离开对象时发生		

表 6-13 列举了 document 对象的键盘事件和它们的使用说明。

表 6-13 document 对象按键事件及其应用

按键事件	使用说明
onKeyDown	用户按下按键时发生
onKeyPress	用户按下按键时发生 onKeyDown 事件,然后产生 onKeyPress 事件,如果用户按住按键不放,则产生一系列 onKeyPress 事件
onKeyUp	用户放开按键时发生
onHelp	用户按下系统定义的帮助键时发生

6.6.4 document 对象的应用案例

例 6.8 页面上有两个文本框,在第 1 个文本框内输入内容后,单击第 2 个文本框,

将在第 2 个文本框内显示第 1 个文本框的内容。实现上述任务的代码 ex6-08.html 清单如下：

```
<html>
<head><title>document 对象应用</title></head>
<body>
 将文字输入文本框 1
<form>
 <input type=text onChange="document.my.elements[0].value=this.value;">
   <! --输入文本框元素[0]=输入的内容//--><br>
</form>
单击文本框 2 显示文本框 1 内容
<form name="my">
 <input type=text
             onChange="document.form[0].elements[0].
             value=this.value;">
   <! --输出文本框元素[0]=输入的内容//-->
</form>
</body></html>
```

ex6-08.html 在浏览器中的运行效果见图 6-11。

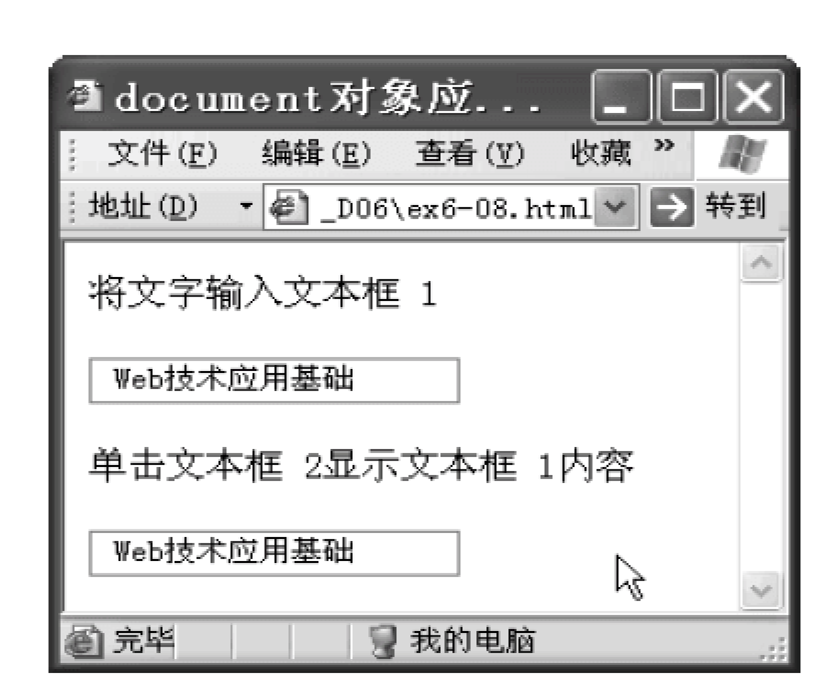


图 6-11　document 对象应用案例

6.7　JavaScript 内置对象

JavaScript 提供了 String(字符串)、Math(数学)、Array(数组)和 Date(日期)内置对象供用户使用。

6.7.1　String 对象

String 对象的应用非常简单，不需要使用关键字 new。例如，var str1 ="Hello World!"即创建了一个名为 str1 的 String 对象。

String 对象的属性只有一个：length，用来统计字符串中字符的个数。例如，上例的 str1.length 的结果值是 12。String 对象的方法见表 6-14。

表 6-14　String 对象的主要方法

方法名称	说　明	范　例
anchor(链接名)	创建 HTML 中的 anchor 标记	str1.anchor("d")
big()	以大写字体显示字符串	str1.big()
small()	以小写字体显示字符串	str1.small()
italic()	以斜体字显示字符串	str1.italic()

续表

方法名称	说明	范例
bold()	以粗体字显示字符串	str1.bold()
blink()	字符串闪烁显示	str1.blink()
fixed()	以固定字高显示字符串	str1.fixed()
fontsize(size)	控制字体大小	str1.fontsize(5)
toLowerCase()	将字符串改变为小写	str1.toLowerCase()
toUpperCase()	将字符串改变为大写	str1.toUpperCase()
indexOf(str,start-position)	在 start-position 查找 str 字符串	str1.indexOf("he",3)
substring(start,end)	返回 start 与 end 位置之间的子串	str1.substring(4,8)

6.7.2 Math 对象

Math 对象包括常用常数和运算，如三角函数、对数函数、指数函数等。Math 对象是一个静态对象，不需要创建具体实例即可使用，例如，var num=Math.sqrt(9)。Math 对象主要属性见表 6-15。

表 6-15 Math 对象主要属性

属性名称	说明	范例
E	常数 e	Math.E=2.718…
LN10	10 的自然对数	Math.LN10=2.302…
LN2	2 的自然对数	Math.LN2=0.693…
LOG2E	以 2 为底 E 的对数	Math.LOG2E=1.442…
LOG10E	以 10 为底 E 的对数	Math.LOG10E=0.434…
PI	圆周率	Math.PI=3.141…
SQRT1_2	0.5 的平方根	Math.SQRT1_2=0.707…
SQRT2	2 的平方根	Math.SQRT2=1.414…

Math 对象主要方法见表 6-16。

表 6-16 Math 对象主要方法

方法名称	说明	范例
sin(),cos()	正弦、余弦	Math.sin(1)=0.841…
asin(),acos()	反正弦、反余弦	Math.asin(1)=1.570…
tan(),atan()	正切、反正切	Math.tan(1)=1.557…

方法名称	说　明	范　例
sqrt()	平方根	Math.sqrt(9)=3
pow(bv,ev)	以 bv 为底的 ev 次方	Math.pow(2,3)=8
abs()	绝对值	Math.abs(-6)=6

6.7.3　Array 对象

1. 语法格式

语法：

数组对象实例名=new Array()

例如：

```
var arr1=new Array()          //创建数组对象实例 arr1,数组大小不定
var arr2=new Array(8)         //创建数组对象实例 arr2,数组长度是 8
```

如果在创建数组对象实例时不给出元素个数,数组的大小在引用数组时确定。数组的下标从 0 开始。

2. Array 对象的属性与方法

Array 对象常用属性是：length 表示数组的长度,等于数组元素的个数。

常用的方法如下。

(1) join()：返回数组中所有元素连接而成的字符串。

(2) reverse()：将数组元素逆转排列,即把数组的第一个元素换成最后一个元素,第二个元素换成倒数第二个元素,依此类推。

(3) sort()：对数组中的元素进行排序。

3. JavaScript 数组对象的特点

(1) 数组元素不要求数据类型相同,可以给一个数组的不同元素赋予不同类型的值。例如：

```
arr1[0]=20          //数值型
arr1[1]="林琳"      //字符串型
arr1[2]=false       //布尔型
```

数组元素可以是数组对象的实例,如果数组元素是数组对象的实例时,得到一个二维数组。例如：

```
var  arr=new Array(8)
```

```
for(i=0;i<8;i++)
    arr[i]=new Array(5)
```

创建了一个 8×5 的二维数组。

(2) 数组的长度可以动态变化。例如,语句 arr=new Array(8)定义了 arr 对象实例的长度为 8,如果希望它的长度增加到 20,只要通过赋值语句 arr[19]=10 就可以了。

6.7.4 Date 对象

JavaScript 的 Date 对象主要用于对日期和时间的操作。它没有属性,但是有多种方法。使用 Date 对象定义日期变量的语法形式如下:

日期对象实例名=new Date()

例如:MyDate=new Date()。

该语句建立了一个日期对象的实例 MyDate,如果没有特别指定时间,将把系统的机内时间放入 MyDate 变量。

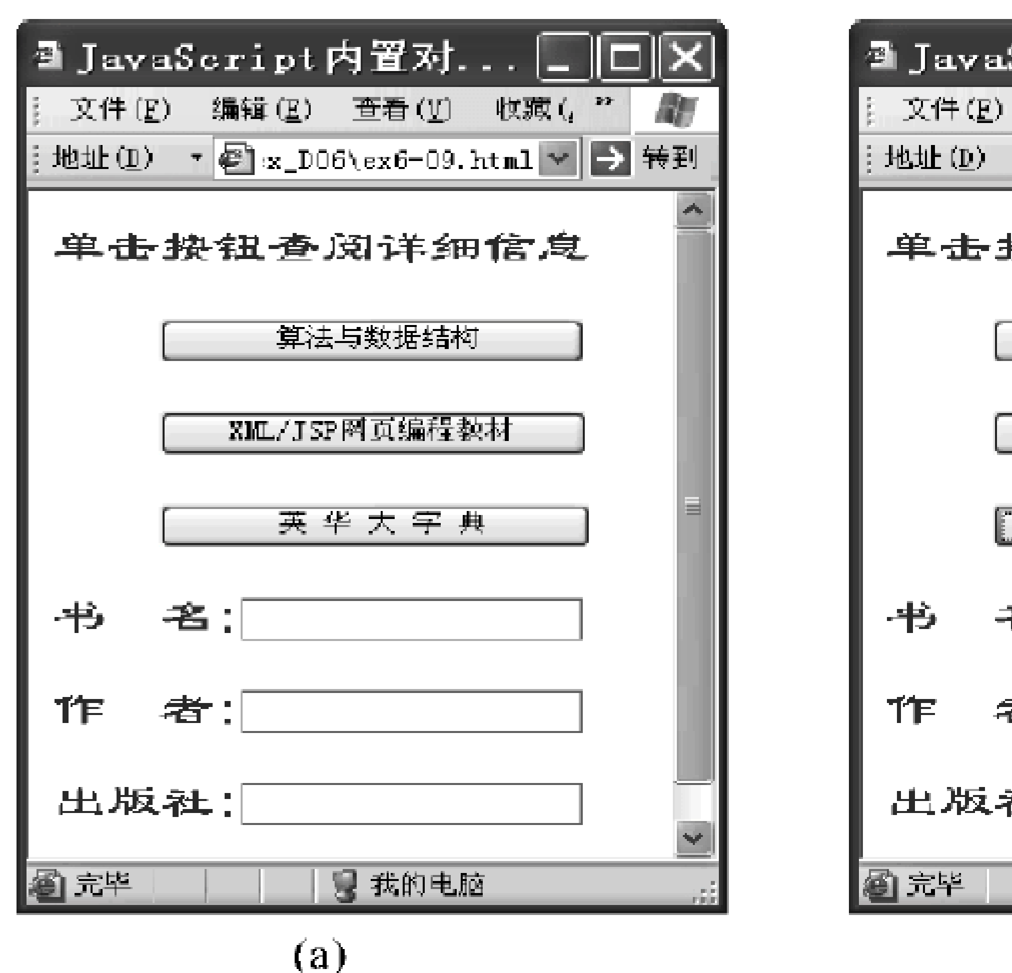
表 6-17 列举了 Date 对象的主要方法和它们的使用说明。

表 6-17 Date 对象的主要方法

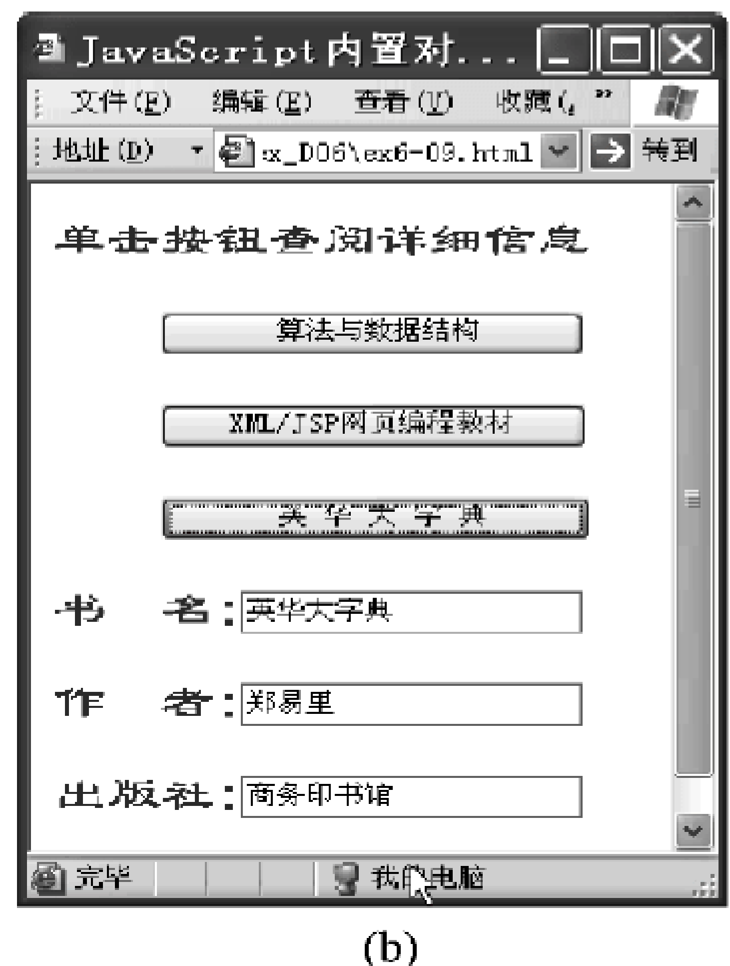
方法名称	说明	范例
getYear()	返回年号	MyDate.getYear()
getMonth()	返回月份数,其值为 0~11,0 代表 1 月	MyDate.getMonth()
getDate()	返回日期,其值为 1~31	MyDate.getDate()
getDay()	返回星期,其值为 0~6,0 表示星期日	MyDate.getDay()
getHours()	返回小时数,其值为 0~23	MyDate.getHours()
getMinutes()	返回分钟数,其值为 0~59	MyDate.getMinutes()
getSeconds()	返回秒数,其值为 0~59	MyDate.getSeconds()
getTime()	返回表示时间的整数,该时间从 1970 年 1 月 1 日午夜开始以毫秒为单位进行计算	MyDate.getTime()
setYear(timevalue)	设置年份,timevalue 为大于 1900 的整数	MyDate.setYear(2008)
setMonth(timevalue)	设置月份数,timevalue 的值为 0~11,0 代表 1 月	MyDate.setMonth(7)
setDate(timevalue)	设置日期,timevalue 的值为 1~31	MyDate.setDate(20)
setDay()	设置星期	MyDate.setDay(5)
setHours(timevalue)	设置小时数,timevalue 的值为 0~23	MyDate.setHours(12)
setMinutes(timevalue)	设置分钟数,timevalue 的值为 0~59	MyDate.setMinutes(30)
setSeconds(timevalue)	设置秒数,timevalue 的值为 0~59	MyDate.setSeconds(30)
setTime()	设置用长整数表示的时间,该时间从 1970 年 1 月 1 日午夜开始以毫秒为单位进行计算	MyDate.setTime(3000)

6.7.5 JavaScript 内置对象应用案例

例 6.9 页面有三个按钮，按钮上显示不同的书名，单击按钮，在下面的文本框中显示该书的信息。页面如图 6-12 所示，图 6-12(a)是打开时的页面，图 6-12(b)是用户选择"英华大字典"后显示的图书信息。



(a)



(b)

图 6-12 JavaScript 内置对象应用案例

实现以上任务的页面代码 ex6-09.html 清单如下：

```
<html><head>
<title>JavaScript 内置对象应用案例</title>
<script language="JavaScript">
<! --
function upBookInfo(titleInfo){
  document.BookForm.BookTitle.value=titleInfo
  document.BookForm.BookAuth.value=this.Auth
  document.BookForm.BookPublisher.value=this.Publisher
}
function Book(title,auth,publisher){
  this.Title=title
  this.Auth=auth
  this.Publisher=publisher
  this.UpInfo=upBookInfo
}
-->
</script>
</head>
<body>
<script language="JavaScript">
```

```
    var Books=new Array()
    Books[0]=new Book("算法与数据结构","严蔚敏 陈文博","清华大学出版社")
    Books[1]=new Book("XML/JSP 网页编程教材","吴艾","北京希望电子出版社")
    Books[2]=new Book("英华大字典","郑易里","商务印书馆")
</script>
<font color="blue" face="隶书" size=5>单击按钮查阅详细信息
<form name="BookForm">
    <input type=button value="算法与数据结构"onClick="Books[0].UpInfo('算法与数据结构')">
<p>
    <input type=button value="XML/JSP 网页编程教材"
                onClick="Books[1].UpInfo('XML/JSP 网页编程教材')"><p>
    <input type=button value=" 英 华 大 字 典 " onClick="Books[2].UpInfo('英华大字
典')"><p>
书 名:<input type="text" name="BookTitle"><p>
作 者:<input type="text" name="BookAuth"><p>
出版社:<input type="text" name="BookPublisher"><p>
</font></form>
</body></html>
```

6.8 JavaScript 应用案例

本节给出 JavaScript 的几个应用案例。

6.8.1 数字钟

例 6.10 制作一个页面,页面上显示"单击此处启动数字钟,统计页面持续时间"。当单击文字时,启动数字钟,用以显示当前时刻和网页的持续时间,页面如图 6-13 所示。

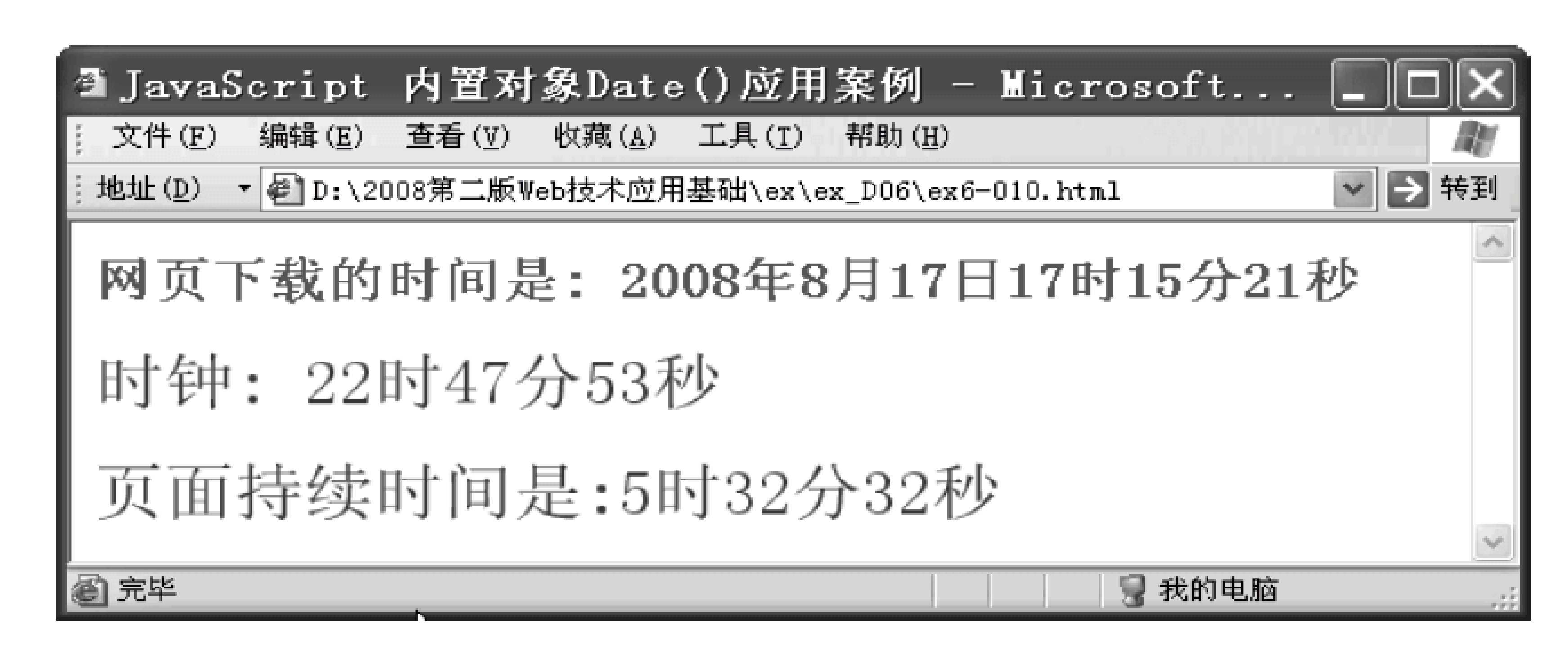


图 6-13 启动数字钟统计网页持续时间

本案例将使用 JavaScript 对象及其内置的日期对象 Date。

实现以上任务的页面代码 ex6-10.html 清单如下:

```
<html><head>
<title>JavaScript 内置对象 Date()应用案例</title>
<script language="JavaScript">
<! --
function Eclock(){
   var MyDate2=new Date()
   var MyTime2=MyDate2.getTime()
   var TimeString2='时钟：'+MyDate2.getHours()+'时'+
                  MyDate2.getMinutes()+'分'+MyDate2.getSeconds()+'秒'
   var MyHours3=0;MyMinutes3=0
   var MySeconds4=Math.round((MyTime2-MyTime1)/1000)
   MyHours3=Math.round((MySeconds4-1800)/3600)
   MyMinutes3=Math.round(((MySeconds4-30)%3600)/60)
   if(MyMinutes3==60)
      MyMinutes3=0
   MySeconds4=Math.round(MySeconds4%3600)
   MySeconds3=Math.round(MySeconds4%60)
   var TimeString3='页面持续时间是：'+MyHours3+'时'+MyMinutes3+'分'+MySeconds3
+'秒'
   Clock.innerHTML=TimeString2+'<P>'+TimeString3
   setTimeout ("Eclock()",1000)
}
-->
</script>
</head>
<body>
<h2><font color="green" face="楷体">
<script language="JavaScript">
<! --
var MyDate1=new Date()
var MyHours1=MyDate1.getHours()
var MyMinutes1=MyDate1.getMinutes()
var MySeconds1=MyDate1.getSeconds()
var MyTime1=MyDate1.getTime()
window.document.write("网页下载的时间是：",
    MyDate1.getYear(),'年',MyDate1.getMonth()+1,'月',
    MyDate1.getDate(1),'日',MyDate1.getHours(),'时',
    MyDate1.getMinutes(),'分',MyDate1.getSeconds(),'秒')
-->
</script></font></h2>
<div id=Clock style="position:absoulte;left:150;top:150;font-family:'楷体';
                    font-size:30;color:#0000FF" OnClick="Eclock()">
单击此处启动数字钟并统计网页持续时间</div>
</body></html>
```

代码说明如下：

① 文件体<body>中的第 5 行：

```
var MyDate1＝new Date()
```

Date 是 JavaScript 内置的日期对象，主要用于对系统的日期和时间进行操作。使用关键字 new 创建了名为 MyDate1 的实例对象，浏览器把本地客户端的时间赋予变量 MyDate1。

② 文件体<body>中第 6 行：

```
MyHours1＝MyDate1.getHours()
```

getHours()是 Date 对象的方法，返回对象中的小时数。该行将 MyDate1 对象中的小时数赋给变量 MyHours1。

③ 文件体<body>中第 9 行：

```
var MyTime1＝MyDate1.getTime()
```

getTime()是 Date 对象的方法，返回对象中表示时间的整数，该时间从 1970 年 1 月 1 日午夜开始以毫秒为单位进行计算。

④ 文件体<body>中第 16 行至第 18 行：

```
<div id＝Clock style＝"position:absoulte;left:150;top:150;font-family:'楷体';
                                    font-size:30;color:#0000FF" OnClick＝"Eclock()">
单击此处启动数字钟并统计网页持续时间</div>
```

在网页上设定了一个 id 是 Clock 的 div 块，该块将由文件头中用 JavaScript 语言书写的 Eclock()函数中的语句引用(代码中的第 19 行)。

```
Clock.innerHTML＝TimeString2＋'<P>'＋TimeString3
```

Clock.innerHTML 表示 id 是 Clock 块中的内容，更新为赋值号右边的值。也就是说，在网页上启动数字钟后，"单击此处启动数字钟并统计网页持续时间"这行文字将由 TimeString2 和 TimeString3 串中的内容替换。

⑤ 文件头中的 Eclock()函数是单击鼠标左键 OnClick 的事件处理程序，启动数字钟，并统计网页的持续时间。

⑥ 代码的第 6 行：

```
var MyDate2＝new Date()
```

使用关键字 new 创建对象 Date 的另一个实例 MyDate2，一个对象可以有多个实例。

⑦ 代码的第 11 行：

```
var MySeconds4＝Math.round((MyTime2-MyTime1)/1000)
```

Math.round()是 JavaScript 内置 Math 对象的 round()方法，把一个数四舍五入为整数。

⑧ 代码的第 20 行：

```
setTimeout ("Eclock()",1000)
```

setTimeout()方法将每隔 1000 毫秒执行一次 Eclock()函数，更新一次数字钟和网页持续时间的数据。

6.8.2 状态栏文字滚动显示

例 6.11 当单击页面时，状态栏的文字滚动显示。

该案例将应用 window 对象的 status 属性，设置状态栏的显示内容。使用 JavaScript 编制 onClick 事件响应程序，使状态栏滚动显示。

完成如上任务的页面代码 ex6-11.html 清单如下：

```
<html><head>
<title>状态栏文字滚动显示</title>
<script language="JavaScript">
<! --
  var ScrText="欢迎学习'Web 技术应用基础'!"
  var LenText=ScrText.length
  var Width=80
  var Pos=1-Width
  function Scroll(){
    Pos++
    var Scroller=""
    if(Pos==LenText){
        Pos=1-Width
    }
    if(Pos<0){
        for(var i=1;i<=Math.abs(Pos);i++)
           Scroller=Scroller+""
        Scroller=Scroller+ScrText.substring(0,Width-i+1)
    }
    else Scroller=Scroller+ScrText.substring(Pos,Width+Pos)
    window.status=Scroller
    setTimeout("Scroll()",16)
}
-->
</script>
</head>
<body onClick="Scroll()">
<div id=StaScroller
style="position:absoulte;left:150;top:150;font-size:20;color:#0000FF" >单击页面注意状态栏的变化</div>
```

```
</body></html>
```

读者可以执行上述程序,观察执行效果。

6.8.3 随机改变页面背景色

例 6.12 随机产生页面的背景色。

完成如上要求的代码 ex6-12.html 清单如下:

```
<html>
<head><title>背景色随机改变</title>
</head>
<body>
<script language=JavaScript>
    var mybool=false
    color_bar=new Array(3)
    for(var i=0;i<color_bar.length;i++){
        while(mybool==false){
          var start_num=(Math.round(Math.random() * 1000))
          if(start_num>255){
             mybool=false
             continue
          }
          color_bar[i]=start_num
          mybool=true
        }
        mybool=false
    }
    var a=color_bar[0].toString(16)
    if(a.length<2){
        a=("0"+a)
    }
    var b=color_bar[1].toString(16)
    if(b.length<2){
        b=("0"+b)
    }
    var c=color_bar[2].toString(16)
    if(c.length<2){
        c=("0"+c)
    }
    var mkcolor=(""+"#"+a+b+c+"")
    document.bgColor=mkcolor
    document.write("<font face='隶书' color='white' size=6>背景色是:"+mkcolor+"</font>")
</script>
```

```
</body></html>
```

ex6-12.html 在浏览器中的显示效果如图 6-14 所示。



图 6-14　背景色随机改变

6.8.4　鼠标跟随

例 6.13　当鼠标在页面上移动时，有一幅图像或一行文字随鼠标移动。完成如上要求的代码 ex6-13.html 清单如下：

```
<html>
<head><title>鼠标跟随</title>
<script language=JavaScript>
<! --
  var x,y
  var CanBool=0
  function canMove(){
    x=document.body.scrollLeft+event.clientX
    y=document.body.scrollTop+event.clientY
    CanBool=1
  }
  function move(){
    if(CanBool){
      ball.style.posLeft=x+20
      ball.style.posTop=y
    }
    setTimeout('move()',100)
  }
-->
</script>
</head>
<body onload="move()" onMouseMove="canMove()">
<font face="隶书" size=5 color=blue>鼠标移动图像跟随效果</font>
<div id="ball" style="position:absolute;left:250px;top:118px;z-index:"6">
   <img src="ball.gif">
</div>
```

```
</body></html>
```

ex6-13.html 在浏览器中的显示效果如图 6-15 所示。



图 6-15　鼠标跟随

6.9　JavaScript 在网上书店中的应用案例

1. 设置页面的大小与格式

在 bookshop 文件夹的 index.jsp 文件中，与页面设置相关的代码段如下：

```
<script language=javascript>
<! -- Begin
function setVariables() {
    imgwidth=50;
    imgheight=50;
    if (navigator.appName == "Netscape") {
        horz=".left";
        vert=".top";
        docStyle="document.";
        styleDoc="";
        innerW="window.innerWidth";
        innerH="window.innerHeight";
        offsetX="window.pageXOffset";
        offsetY="window.pageYOffset";
    }
    else {
        horz=".pixelLeft";
        vert=".pixelTop";
        docStyle="";
        styleDoc=".style";
        innerW="document.body.clientWidth";
        innerH="document.body.clientHeight";
```

```
            offsetX="document.body.scrollLeft";
            offsetY="document.body.scrollTop";
        }
}
function checkLocation() {
    objectXY="branding";
    var availableX=eval(innerW);
    var availableY=eval(innerH);
    var currentX=eval(offsetX);
    var currentY=eval(offsetY);
    x=availableX-(imgwidth+60)+currentX;
    y=availableY-(imgheight+40)+currentY-300;
    evalMove();
    setTimeout("checkLocation()",10);
}
function evalMove() {
    eval(docStyle + objectXY + styleDoc + horz + "=" + x);
    eval(docStyle + objectXY + styleDoc + vert + "=" + y);
}
// End -->
</script>
<link href="maincss.css" rel="stylesheet" type="text/css">
<body onload="setVariables(); checkLocation();">
        <div align="center">
        ...
</body></html>
```

2. 用户登录功能 login.jsp 代码

用户登录功能包括客户端验证与服务器端验证，与服务器端验证相关的代码将在后续章节讲解。login.jsp 程序中的客户端验证代码段如下：

```
<script language="javascript">
<! --
function CheckSubmit()
  {
    if( document.loginform.userid.value == "" )
        { alert("请输入用户名!"); document.loginform.userid.focus(); return false; }
    if( document.loginform.password.value == "" )
        { alert("请输入密码!"); document.loginform.password.focus(); return false; }
    if(document.loginform.userid.value.indexOf("'") ! = -1)
        { alert("用户名不能包含单引号、空格等字符!");
          document.loginform.userid.focus(); return false; }
    return true;
  }
```

```
</script>
<link href="maincss.css" rel="stylesheet" type="text/css">
<table width="100%" border="0" cellpadding="0" cellspacing="0" bgcolor="#f6f6f6" class=
"td">
  <form name="loginform" action="index.jsp? op=login" method="post" >
    <tr>
      <td colspan="3"><div align="left">
        <img src="images/login_r1.jpg" width="158" height="43"></div></td>
    </tr>
    <tr>
      <td width="8"> </td>
      <td width="25%">用户名</td>
      <td width="75%">
        <input name="userid" type="text" class="formtext" size="12"></td>
    </tr>
    <tr>
      <td> </td>
      <td>密码</td>
      <td>
        <input name="password" type="password" class="formtext" size="12">
      </td>
    </tr>
    <tr>
      <td> </td>
      <td colspan="2">
        <input name=" loginbutton" type =" submit" value =" 登 录 " onClick =" return
        CheckSubmit();">
        新用户<a href="register.jsp">注册</a> </td>
    </tr>
  </form>
</table>
```

习题、上机练习与实训 6

一、习题

1. 什么是脚本语言？它的功能是什么？
2. 客户端脚本和服务器端脚本各自的功能是什么？
3. 如何将 JavaScript 嵌入 HTML 页面？请写出它们的关键语句。
4. 如何将 JavaScript 调入 HTML 文档？请写出它们的关键语句。
5. window 对象的组成结构是怎样的？
6. window 对象的主要属性和方法是什么？举例说明它们的应用方法。
7. window 下层最主要的对象是什么？

8. document 对象的主要属性、方法和事件是什么？请举例说明它们的使用方法。

9. document 对象的按键事件在什么时候起作用？

10. JavaScript 中的函数如何定义，如何调用？

11. JavaScript 如何创建对象？如何访问所创建对象的方法和属性？

12. JavaScript 主要应用哪几个接口元素？如何使用？

二、上机练习

1. 使用<script>标记把一小段脚本程序嵌入或调入 HTML 页面，并在浏览器中显示它的结果。

2. 使用 JavaScript 制作一个跑马灯。

3. 使用 JavaScript 编制一段代码完成以下功能：

(1) 要求输入一个姓名。

(2) 用确认框检查输入是否正确(是否为合法输入字符，位长是否合理等)。

(3) 根据输入给出相应的提示。

4. 页面上有一幅图像，在状态栏显示有关图像的说明。单击图像时，换成另一幅图像，同时状态栏的内容也做相应的变更。

5. 制作页面，实现鼠标移动时，一行文字跟随鼠标移动。

6. 制作一个页面，页面上有两个文本框和提交按钮，在文本框中输入信息后，用鼠标单击提交按钮后，将显示文本框中输入的内容。

7. 在客户端验证用户输入信息，如果输入正确，允许链接到网站；如果不正确，禁止链接，要求用户界面友好。

8. 制作一个页面，页面的背景色可以随机变化。

9. 制作一个数字钟，根据网页下载持续时间进行收费。

10. 制作一个打猎游戏，页面上有几个猎物，或在飞或在跑，用鼠标追赶猎物，追上后用鼠标单击猎物，弹出信息框说明猎物被击中，同时猎物消失。

11. 为某单位的主页制作标题，使标题具有动态效果。

三、实训课题

1. 应用 JavaScript 制作一个计算器，可以对整数和小数进行加、减、乘、除运算，计算器的界面如图 6-16 所示。

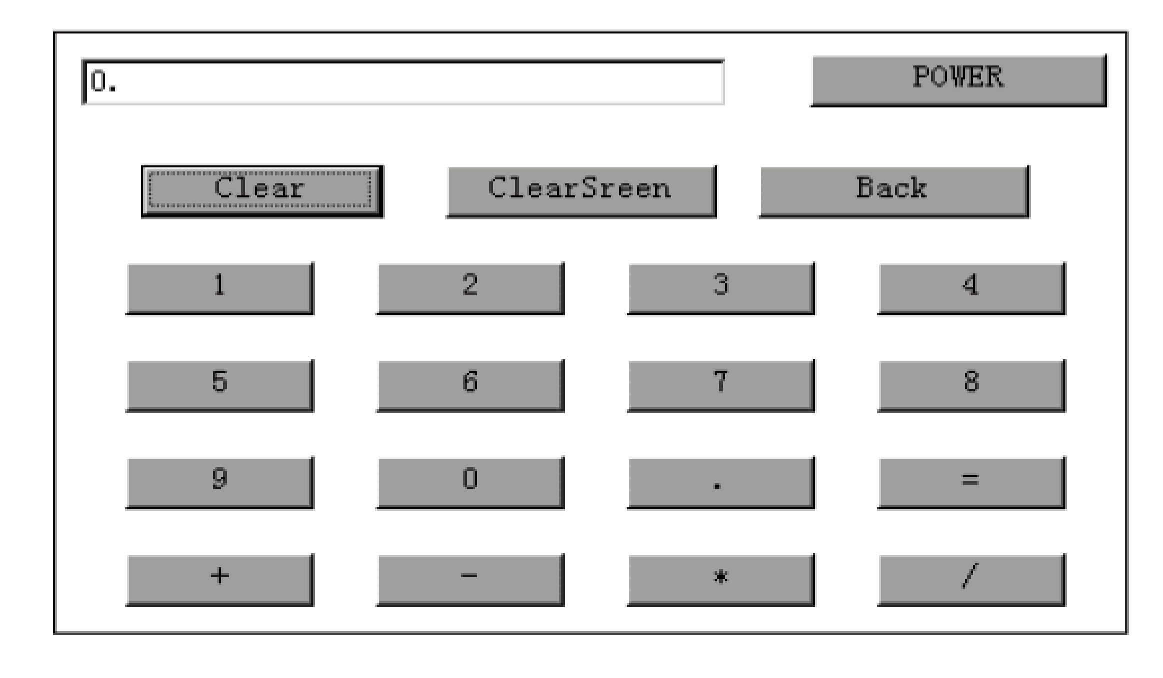


图 6-16　计算器界面

2. 为某企业网站设计并制作一个业务查询栏目。输入正确的用户名及口令后，允许进行查询。请读者完成用户需求分析，确定查询的项目，设计并规划栏目的界面，并使用 JavaScript 脚本语言完成控件之间的响应任务和计算任务（如果存在计算任务的话）。

3. 某银行网站希望增加一个栏目，在界面中输入本金、存款方式和存款时间后，即可得到到期后的本金和利息之和共有多少，以利于客户选择存款方式。完成该栏目的设计与制作。

4. 制作一个扑克牌的 21 点游戏，游戏开始随机发 4 张牌，通过加、减、乘、除使答案为 21。设计该游戏并完成它的制作。

第 3 篇　JSP Web 数据库应用开发

JSP 技术是开发 Web 数据库应用的先进工具。它是基于 JavaServlet 以及整个 Java 体系的 Web 开发技术。应用 JSP 技术可以开发动态、高性能、安全和跨平台先进的 Web 应用系统。通过第 3 篇的学习，读者将掌握 JSP Web 数据库应用开发技术。第 3 篇将主要介绍以下内容：

第 7 章　JSP 运行机制与基本语法
第 8 章　JSP 内置对象
第 9 章　基于 JSP 的 Web 数据库应用开发
第 10 章　网上书店的实现
第 3 篇所有案例在“Windows XP＋Tomcat＋SQL Server 个人版”环境上调试通过。

第 7 章　JSP 运行机制与基本语法

7.1　JSP 技术概述

第 7 章案例应存放在 Tomcat 发布目录下，本章案例存放在D:\tomcat5.5\webapps\ex_D07 目录下。

7.1.1　JSP 应用示例

例 7.1　第一个 JSP 程序，制作一个简单的 JSP 页面，在浏览器中输出 3 行由小变大的文字，制作步骤如下。

1. 在记事本中输入 JSP 源代码

打开记事本，输入以下代码，文件是文本格式的，编辑完成后，以.jsp 后缀保存在D:\tomcat5.5\webapps\ex_D07 目录下。由于 JSP 是基于 Java 的，所以文件名区分大小写。ex7-01.jsp 的源代码如下：

```
<%@ page contentType="text/html; charset=GB2312"%>
<html>
<head><title>第一个 JSP 页面</title>
</head>
<body>
<%--JSP 中的注释语句--%>
<center>
<%="<font size=4 color=red>字体由小变大显示</font>"%>
</center><br><hr><br>
<div align="center">
<%
   //使用 Java 语言的 for 循环语句控制输出字体的大小
   for( int i=4; i>1; i-- )
      out.println( "<h" + i + ">Web 技术应用基础</h" + i + ">" );
%>
```

```
</div>
</body></html>
```

2. 在浏览器中显示结果

应用发布方式查看结果。打开浏览器，在浏览器地址栏目中输入 ex7-01.jsp 文件的 URL：http://127.0.0.1:8080/ex_D07/ex7-01.jsp。该段代码在浏览器中的显示如图 7-1 所示。



图 7-1　字体由小变大

3. 经服务器端作用后在客户端见到的源代码

在客户端浏览器菜单中选择“查看”→“源文件”，观察经服务器端作用之后，在客户端显示的代码如下：

```
<html>
<head><title>第一个 JSP 页面</title>
</head>
<body>
<center>
<font size=4 color=red>字体由小变大显示</font>
</center><br><hr><br>
<div align="center">
<h4>Web 技术应用基础</h4>
<h3>Web 技术应用基础</h3>
<h2>Web 技术应用基础</h2>
</div>
</body></html>
```

这段代码与服务器端的代码是不同的，在服务器端的 ex7-01.jsp 文档中的脚本：

```
<%
    for( int i=4; i>1; i-- )
    out.println( "<h" + i + ">Web 技术应用基础</h" + i + ">" );
```

```
%>
```

经服务器端编译执行后，在客户端形成标准的 HTML 语句：

```
<h4>Web 技术应用基础</h4>
<h3>Web 技术应用基础</h3>
<h2>Web 技术应用基础</h2>
```

客户端的用户见不到服务器端运行的源代码，只能看到运行后的标准 HTML，这将使源代码不外泄，有助于开发者保护自己的知识产权。

7.1.2 JSP 运行机制

JSP 是服务器端技术，在服务器端 JSP 引擎解释 JSP 代码，然后将结果以 HTML 页面形式发送到客户端，在客户端的用户是看不到 JSP 代码的。

以代码 ex7-01.jsp 为例，说明 JSP 程序的运行过程。

① 当服务器上的一个 JSP 页面第一次被请求执行时，服务器上的 JSP 引擎解析页面，并生成一个 Java 源文件 *.java。读者可以打开 Tomcat 的工作目录：

```
D:\Tomcat 5.5\work\Catalina\localhost\ex_D07\org\apache\jsp
```

在该目录下存放 JSP 引擎工作过程中产生的 *.java 和 *.class 临时文件。找到对应的 ex7_002d01_jsp.java 文件，这就是 JSP 引擎解析 ex7-01.jsp 时得到的 Java 源程序，如果读者有兴趣，可以用文本编辑器打开该文件查看。

② 然后把该 Java 文件编译生成 Java class 字节码文件 *.class。这个 class 文件就是 Servlet，Servlet 引擎像处理其他所有的 Servlet 一样处理该 class 文件。字节码文件的任务是：

- 把 JSP 页面中的 HTML 标记送到客户端浏览器执行显示。
- JSP 标记、数据和方法声明、Java 程序段在服务器解释执行，把需要显示的结果嵌入 HTML 页面送客户端浏览器显示。
- 由服务器计算 Java 表达式，把计算结果转化为字符串，送交客户端浏览器显示。在 Tomcat 的工作目录下，读者可以看到 ex7_002d01_jsp.class 文件。

③ Servlet 引擎载入 class 文件开始执行。

④ Servlet 执行完成后，把结果返回给发出请求的客户。

过程①和②只有在页面首次执行或升级 JSP 时才发生，Servlet 引擎只在服务器重新启动后产生第一个请求时才执行过程③；class 载入器仅载入 class 文件一次，并且在 Java 虚拟机的运行期间内均有效可行。过程④的执行效率与数据量有关。由于在 JSP 程序的执行过程中预先生成了.class 文件，所以同其他技术相比，JSP 的运行速度是较快的。

7.1.3　JSP 的特点

JSP 的主要特点如下。

1．把内容的生成和显示分离

使用 JSP 技术可以把界面的开发与程序逻辑的开发分离开。Web 开发人员使用 HTML 标记来设计界面，使用 JSP 标记或脚本生成页面上的动态内容。JSP 技术使得开发人员的分工更加明确，页面设计者可以修改内容的显示而不影响逻辑，应用程序的开发者修改逻辑而不影响内容显示。

2．生成可重用的组件

绝大多数 JSP 页面依赖于可重用的、跨平台的组件（JavaBean 或 Enterprise JavaBean）来执行应用程序要求的复杂处理。开发人员可以共享和交换组件，或把这些组件提供给更多的用户使用。基于组件的开发方法有效地提高了应用程序的开发效率，加速了项目的总体开发进程。

3．应用标记简化页面的开发

JSP 技术封装了许多功能，这些功能是生成与 JSP 相关的 HTML 或 XML 的动态内容时所需要的。标准的 JSP 标记能够访问和实例化 JavaBean 组件，设置或检索组件的属性，下载 Applet，以及执行使用其他方法难以实现的功能。

JSP 标记具有可扩充性，允许开发者扩展 JSP 标记，开发人员也能够定制常用功能标记库。第三方或其他开发人员也可以创建自己的标记库。由于页面制作者可以使用标记库中的功能，大大减少了对脚本语言的依赖，并降低了页面制作的复杂度。

4．具有 Java 特点

由于 JSP 页面的内置脚本是基于 Java 语言的，而且所有的 JSP 页面都被编译成 Java Servlet，所以 JSP 具有 Java 的特点，如健壮的存储管理、安全性、跨平台特性、“一次编写，各处运行”等。

7.1.4　JSP 页面结构

由例 7.1 可以看出 JSP 文件由两部分组成：一部分是＜％…％＞标记以外的部分；另一部分是在＜％…％＞标记内的代码，标记内的代码即为 JSP 代码。JSP 代码由 JSP 引擎处理。

在 HTML 文件中嵌入 Java 程序段（scriptlet）和 JSP 标记（tag），就组成了 JSP 页面（＊.jsp）。Web 服务器遇到 JSP 页面时，先执行嵌入的 JSP 程序段，然后将运行结果与其他的 HTML 组合到一起返回给客户。Java 程序段可以操作数据库、重新定向页面和

发送 E-mail 等。

7.2 JSP 基本语法

7.2.1 JSP 页面组成

JSP 常用的语句主要有以下 6 种类型。

① 注释：<%--注释内容--%>

② 声明：<%！预定义内容%>

③ 表达式：<%＝表达式%>

④ 脚本段 Scriptlet：<%代码%>

⑤ 指令：<%@指令%>

⑥ 动作：<jsp:动作>

7.2.2 注释

注释增加了程序的可读性与可维护性。JSP 中的注释分为两种：一种是发送到客户端，在客户端可见的注释，称为 HTML 注释；另一种是只存在于服务器端，在客户端不可见的注释，称为隐藏注释。

1. 注释示例

为程序注释时，有些内容是可以传递到客户端的，有些是不希望被客户见到的。为此，可以分别用 HTML 注释和 JSP 注释。

例 7.2 在例 7.1 中增加注释并作适当的修改后，ex7-02.jsp 的源代码如下：

```
<%@ page contentType="text/html; charset=GB2312"%>
<html>
<head><title>JSP 的注释语句</title>
</head>
<body>
<!--HTML 注释-->
<!--页面访问时间：<%=(new java.util.Date()).toLocaleString()%>-->
<center>
<%--服务器端注释--%>
<%="<font size=4 color=red>字体由小变大显示</font>"%>
</center><br><hr><br>
<div align="center">
<%
   //控制文字大小的循环语句
```

```
    for( int i=4; i>1; i-- )
      out.println( "<h" + i + ">Web 技术应用基础</h" + i + ">" );
%>
</div></body></html>
```

程序运行结果见图 7-1。观察经服务器端作用之后，在客户端见到的 ex7-02.jsp 的源文件，具体如下：

```
<html>
<head><title>JSP 的注释语句</title>
</head>
<body>
<! --HTML 注释-->
<! --页面访问时间：2008-8-19 10:22:37-->
<center>

<font size=4 color=red>字体由小变大显示</font>
</center><br><hr><br>
<div align="center">
<h4>Web 技术应用基础</h4>
<h3>Web 技术应用基础</h3>
<h2>Web 技术应用基础</h2>

</div></body></html>
```

可以看出 ex7-02.jsp 与 ex7-01.jsp 在浏览器中显示的结果是一样的，在 ex7-02.jsp 中的注释有的可在客户端见到，有些却看不到。

2. HTML 注释

JSP 引擎对 HTML 注释不作任何解释，直接送交客户端浏览器，在客户端浏览器中可以查看到 HTML 注释。HTML 注释的语法规则如下：

<! --注释<%=表达式%>-->

注释在“<! --”和“-->”标记之间，可以在注释中包含各种合法的表达式。

例如，<! --<%=str %>-->，以及 ex7-02.jsp 中的注释语句：<! --HTML 注释-->。

如果在注释中使用了表达式，所有嵌入的 JSP 代码仍在服务器端编译执行，并将执行结果返回客户端的源代码中。例如，ex7-02.jsp 中的注释语句：

```
<! --页面访问时间：<%=(new java.util.Date()).toLocaleString()%>-->
```

在客户端见到的源代码是：

```
<! --页面访问时间：2008-8-19 10:22:37-->
```

3. JSP 注释

JSP 注释在客户端是见不到的，故也称隐藏注释。隐藏注释写在 JSP 代码中，是为 JSP 代码做的注释，它们不发送到客户端，所以在客户端的源文件中见不到隐藏注释。隐藏注释的语法规则如下：

<%--注释--%>

JSP 引擎在编译时将忽略"<%--"和"--%>"之间的语句。隐藏注释是给编程人员看的，属于内部资料，它们既不在客户端浏览器中显示，也不能在客户端的"查看源文件"中看到。例如，ex7-02.jsp 中的注释语句：

```
<%--服务器端注释--%>
```

在客户端的源文件中被空白行代替。

7.2.3 声明

声明语句声明将要在 JSP 文件中用到的变量和方法，变量类型包括 Java 的基本类型及其类对象。在"<%!"和"%>"标记之间声明变量和方法。在这两个标记之间声明的变量在整个 JSP 页面有效。当 JSP 引擎将 JSP 页面转译成 Java 文件时，把这些变量生成为类的成员变量，它们的内存空间在服务器关闭后才被释放。当多个用户请求同一个 JSP 页面时，JSP 引擎为每个用户启动一个线程，这些线程由 JSP 引擎管理，并由这些用户共享 JSP 页面的成员变量。应用用户共享成员变量的特点，可以制作计数器等应用。

1. 声明的语法规则

声明的语法规则如下：

<%! 声明;[声明;]…%>

例如：

```
<%! int i=6;%>
<%! int a,b,c;double d;%>
<%! Circle a=new Circle(6.0);%>
```

2. 使用注意事项

使用时需要注意：

① 可以一次声明多个变量和方法，必须以","分开，以";"号结尾。

② 一个声明只在一个页面有效。最好把每个页面都要用到的声明写成一个单独的文件，然后用<%@ include %>指令或<jsp:include>动作包含进来。

③ 可以直接使用在<%@ page %>中包含进来已经声明了的变量和方法，不需要

对它们重新声明。

3. 应用示例

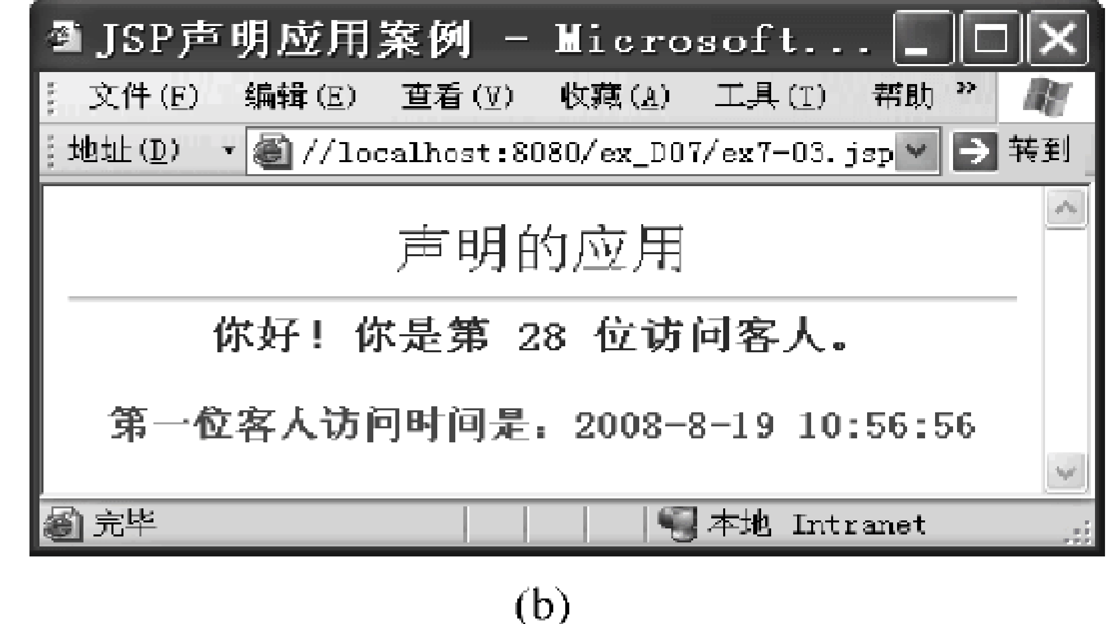
例 7.3 ex7-03.jsp 声明了一个 int 类型的变量 num，用来统计访客人数，两个 String 变量和一个时间对象。因为 JSP 声明不产生任何输出，所以需要和 JSP 表达式及 JSP 脚本结合起来使用，把它们输出到页面上。当有多个用户访问该页面时，所有用户共享变量 num，读者可以在多个浏览器窗口打开该页面观察该共享现象。ex7-03.jsp 源代码如下：

```
<%@ page  import="java.util.*"%>
<%@ page contentType="text/html; charset=GB2312"%>
<html><head><title>JSP 声明应用案例</title>
</head>
<body>
  <center>
    <%="<font size=5 color=blue>声明的应用</font>"%>
  </center><hr>
  <%! int num=0;%>
  <%! String str1,str2;%>
  <% str1="你好！ 你是第  "; str2="  位访问客人。";%>
  <% num++;%>
  <%! Date MyDate=new Date();%>
  <div align="center"><font size="4" color=blue><b>
    <%=str1%><%=num%><%=str2%></font><p>
    <font color=green>
      第一位客人访问时间是：<%=MyDate.toLocaleString()%>
    </font></b></div>
</body></html>
```

ex7-03.jsp 在浏览器中显示如图 7-2 所示，图 7-2(b)是第 28 位客户访问时显示的页面，num 变量被多个用户共享。



(a)



(b)

图 7-2 声明的应用

7.2.4 表达式

JSP 的表达式是由变量和常量组成的算式，它将 JSP 生成的数值嵌入 HTML 页面，用来直接输出 Java 代码的值。

1. 表达式的语法规则

在“＜％＝”和“％＞”之间插入 JSP 表达式。由服务器计算表达式的值，并将计算结果以字符串形式送到客户端浏览器显示。如果表达式的值是字符串，该表达式的值将直接显示在页面上，否则将表达式的值转换为字符串，在页面上显示。由于表达式已经被转化为字符串，所以可以在一行文本中插入表达式。表达式中可以使用 request、response、out、session、application、config 和 PageContext 等 JSP 内部对象。

语法：

＜％＝表达式％＞

例如 ex7-01. jsp 中的＜％＝″＜font size＝4 color＝red＞字体由小变大显示＜/font＞″％＞。

2. 应用注意事项

应用时需注意以下几点：

① 不能用“；”号做表达式的结束标志，但是同样的表达式在 Scriptlet 中需要用分号作为结束符。

② “＜％＝”是一个完整的标记，中间不能有空格。

③ 表达式元素包括任何在 Java 语言规范中有效的表达式。

④ 表达式可以成为其他 JSP 元素的属性值。一个表达式可以由一个或多个表达式组成，按从左到右的顺序求值。

应用示例见例 7.3 代码 ex7-03. jsp 中的语句：

```
＜％＝str1％＞＜％＝num％＞＜％＝str2％＞
＜％＝MyDate. toLocaleString()％＞
```

7.2.5 JSP 脚本段

1. JSP 脚本元素语法规则

JSP 脚本段可以包含任意行的合法的脚本语句(Scriptlet)，脚本段是一个代码片段，在服务器处理请求过程中被执行。JSP 脚本代码界定在“＜％”和“％＞”之间，在界定标记之间的内容在服务器端被脚本引擎编译执行，执行结果重新嵌入 HTML 后一起发送到浏览器端。

一个 JSP 页面可以有多个脚本段，这些脚本段将被 JSP 引擎顺序执行。在一个脚本

段中声明的变量是JSP页面的局部变量,它们在后续的JSP页面的脚本段和表达式中有效。当JSP引擎把JSP页面转译成Java文件时,把脚本段的变量生成为某个类的方法中的变量,也就是局部变量。当脚本段被调用时,局部变量被分配内存空间;调用结束后,局部变量占有的内存空间被释放。在多个用户请求同一个页面时,JSP引擎为每个用户启动一个线程,为不同用户的局部变量分配不同的内存空间,所以某个用户对局部变量的操作,不会影响另一个用户的局部变量。

语法:

```
<%代码%>
```

例如,例7.1的代码ex7-01.jsp中的语句:

```
<%
    //控制文字大小的循环语句
    for( int i=4; i>1; i-- )
      out.println( "<h" + i + ">Web技术应用教程</h" + i + ">" );
%>
```

就是一个脚本段,变量i为局部变量,用来控制字体的大小。

2. 脚本元素的功能

脚本元素可以和页面的静态元素(例如HTML)组合在一起生成动态页面。一个Scriptlet能够包含多个JSP语句、方法、变量和表达式。脚本段的主要功能如下:

① 声明将要用到的方法和变量。

② 编写JSP表达式。

③ 编写JSP语句,如果使用Java语言,这些语句必须遵从Java语言规范。

④ 使用任何隐含的对象和任何用<jsp:useBean>声明过的对象。

⑤ 填写任何文本和HTML标记。

如果脚本段有需要显示的内容,则把这些内容存放在out对象中,输出到页面上显示。例如,ex7-01.jsp中的语句:

```
out.println( "<h" + i + ">Web技术应用基础</h" + i + ">" )
```

当i等于2时,相当于out.println("<h2>Web技术应用基础</h2>"),把文字“Web技术应用基础”输出到页面上。

7.2.6 JSP基本语法应用案例

例7.4 根据Web服务器系统的时间,显示不同时间段的问候。ex7-04.jsp代码清单如下:

```
<%@ page contentType="text/html;charset=gb2312" import="java.util.*" %>
<html>
<head><title>jsp基本语法应用案例 </title>
```

```
</head>
<body>
<center><font color=blue size=6 face="隶书">
<%
    Date today=new Date();
    int hours=today.getHours();
    int minute=today.getMinutes();
    if(hours>=0 && hours<12){
        out.println("朋友们,早上好!");
    }
    else if(hours>=12 && hours<19){
        out.println("朋友们,下午好!");
    }
    else
        out.println("朋友们,晚上好!");
%>
</font></center>
</body></html>
```

ex7-04.jsp 使用 new 关键字创建了一个日期对象,把系统的时间赋予 today 变量,根据不同的时间段,输出不同的问候语发送到浏览器。

在浏览器中显示效果如图 7-3 所示。

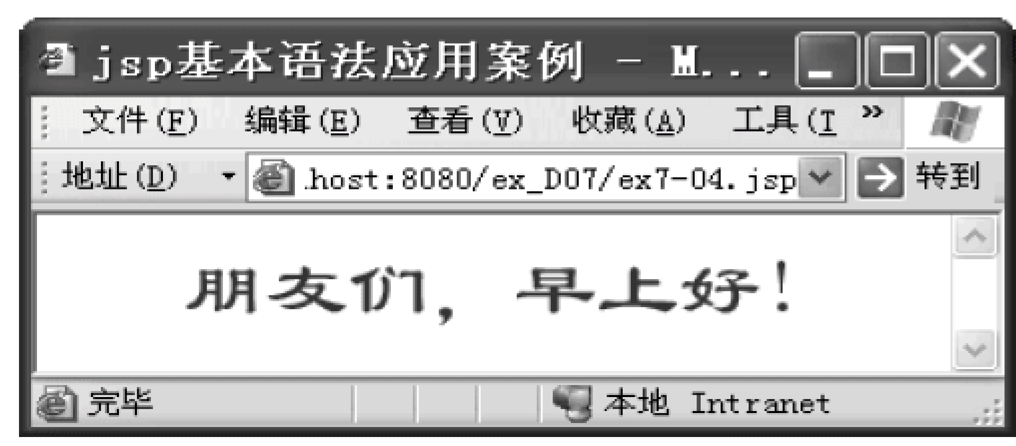


图 7-3 显示不同时间的问候

7.3 JSP 指令

7.3.1 JSP 指令功能

JSP 指令是一些特殊的 JSP 语句,它是为 JSP 引擎设计的,通知 JSP 引擎如何处理 JSP 页面,不直接产生任何可见输出。JSP 指令将执行 JSP 代码有关的信息传递给 JSP 引擎。例如,缓冲器和线程信息、错误处理信息,脚本语言的指定以及所有的扩展,包含的外部文件(被包含的文件可能是另一个 JSP 文件),指出页面可以调用的标记库等。

JSP 指令由"<%@"开始,以"%>"结束,例如,<%@ page language="java" %>。两个常用的 JSP 指令是:include 和 page。其他如 import、errorPage 和 session 也是 JSP 常用指令。应用举例如下:

<%@ include file="filename.jsp" %>,指出被包含的文件。

<%@ page import="java.util.Date" %>,指定在哪里找到支持的 Java 类。

<%@ page errorPage="errorPage.jsp" %>,指出如果发生 Java 异常事件,将信息发送到哪个异常处理页面。

<%@ page session="true" %>,指示是否需要为使用者管理会话期的信息。

7.3.2 include 指令

1. include 指令语法规则

<%@ include file="文件 URL"%>

2. include 指令功能

include 指令称为文件加载指令，其功能是在 JSP 文件编译时，加载需要嵌入的文本或代码，它把文件嵌入当前位置后合并成一个新的 JSP 页面，再由 JSP 引擎转译成 Java 文件。被嵌入的文件可以是 JSP 文件、HTML 文件或其他文本文件。被嵌入的文件必须是可以访问和可使用的。"文件 URL"是要嵌入文件的 URL，一般是指相对路径，不需要指明端口、协议和域名。如果被嵌入的文件与当前 JSP 页面位于同一个 Web 服务目录，"文件 URL"以文件名或路径名开始。如果被嵌入的文件更改了，当前 JSP 文件将被重新编译。嵌入新文件后，必须保证合并成的新 JSP 文件符合 JSP 页面的语法规则。例如，JSP 页面不允许使用 page 指令对除 import 指令外的一个属性多次赋值，如果有两个文件的 page 指令对同一个属性赋值，那么在使用 include 指令时就会出错。

在开发 Web 应用时，如果有多个页面含有相同的功能，可以把这些相同功能的内容放在一个文件中，由使用这些功能的页面使用 include 指令将该文件加载进来，如此可以提高 Web 应用的开发效率，并便于应用系统管理与维护。

3. include 指令应用案例

例 7.5 文件 ex7-05_1.jsp 输出系统的日期和时间，并用 include 指令把它嵌入 ex7-05.jsp 文件，在页面上显示。

ex7-05.jsp 代码清单如下：

```
<%@ page contentType="text/html; charset=gb2312"%>
<html>
<head><title>include 指令应用</title>
</head>
<body><center>
<font size=5 face="隶书" color=blue>include 指令应用</font><br><hr>
<font size="4" color=red><b>
现在的日期和时间是：<br>
<%@ include file="ex7-05_1.jsp"%>
</font></center>
</body></html>
```

ex7-05_1.jsp 代码清单如下：

```
<%@ page language="java" import="java.util.*"%>
<%=(new Date()).toLocaleString()%>
```

运行程序 ex7-05.jsp 在浏览器中显示如图 7-4 所示。

在客户端见到的源文件如下：

```
<html>
<head><title>include 指令应用</title>
</head>
<body> <center>
<font size=5 face="隶书" color=blue>include 指令应用</font><br><hr>
<font size="4" color=red><b>
现在的日期和时间是：<br>
2008-8-19 18:45:00
</font></center>
</body></html>
```



图 7-4 include 指令应用

在客户端的源文件中，见不到 ex7-05_1.jsp 源代码，在 ex7-05.jsp 文件的<%@ include file="ex7-05_1.jsp"%>位置嵌入了 ex7-05_1.jsp 文件的执行结果：2008-8-19 18:45:00。

7.3.3 page 指令

1. page 指令功能

page 称为页面指令，用来定义整个 JSP 文件要使用的属性和属性值，在 JSP 程序编译时将这些属性和属性值传递给 JSP 引擎。page 指令可以用来指定所使用的脚本语言、JSP 代表的 Servlet 实现的接口，导入指定的类及软件包等。

2. page 指令的语法规则

<%@ page 属性 1="属性值 1"属性 2="属性值 2"… %>

或

```
<%@ page
    [language="java"]
    [import="{package.class|package.*},…"]
    [extends="package.class"]
    [session="true|false"]
    [buffer="none|8kb|sizekb"]
    [autoFlush="true|false"]
    [isThreadSafe="true|false"]
    [info="text"]
    [errorPage="relativeURL"]
    [contentType="mimeType [;charset=characterSet ] "|"text/html;charset=ISO-8859-1"]
    [isErrorPage="true|false"]
```

%>

<%@ page %>指令作用于整个JSP页面，几乎在所有的JSP页面的开头都可以见到page指令。在一个页面中可以使用多个<%@ page%>指令，分别描述不同的属性，每个属性只能用一次，但是import属性可以多次使用。<%@ page %>指令区分大小写，它的属性与属性的使用见表7-1。

表7-1　page指令的属性

属性名	使　用　说　明	范　　例
language	定义要使用的脚本语言，默认值是“java”，也是目前唯一有效的设定值，故可以不设	language="java"
import	用来导入在后面的代码中将会用到的一个或多个包/类。只有import可以在同一页面为一个属性多次设值。多个属性值用“,”号分开。JSP默认的import属性有："javax.servlet.jsp.*"、"javax.servlet.*"、"java.lang.*"、"javax.servlet.http.*"	import="java.util.*"
extends	定义JSP编译时需要继承的Java父类	extends="package.class"
session	设置在当前页中是否允许session操作，默认值是true。如果设为false，则不能使用session对象	session="true"
errorPage	指定一个JSP页面，所有未被捕获的异常均由该页面处理	errorPage="relativeURL"
isErrorPage	设置当前页是否可以作为其他JSP页面的异常处理页面，如果被设置为true，就能使用exception对象，默认值是“false”	isErrorPage="true"
contentType	定义输出的MIME类型和JSP文件的字符编码，默认值是：text/html，默认字符集是：ISO 8859—1。如果需要输出汉字，则将字符集设为GBK或GB2313	contentType="text/html; charset=GB2312"
isThreadsafe	设置JSP是否支持多线程。如果取值为true，JSP能够同时处理多个用户的请求；如果设为false，JSP只能一次处理一个请求。默认值是true	isThreadsafe="true"
buffer	指定JspWriter类型预定义对象out缓冲区的大小。取值buffer="none|8kb|sizekb"，none则没有缓冲区。默认为8KB	buffer="none"
autoFlush	设置缓冲区被填满时是否自动刷新，取值为autoFlush="true|false"，默认值true。如果取值true，在缓冲区溢出时，自动输出；如果取值false，在缓冲区溢出时，抛出一个异常。一般不设为false	autoFlush="true"
info	用于定义一个将加入到已编译成功的页面中的字符串	info="text"

page指令实例如下：

```
<%@ page language="java"%>                  //使用的脚本语言是Java
<%@ page import="java.util.*" %>            //导入java.util包
<%@ page session="true" %>
```

```
<%@ page contentType="text/html ; charset=gb2312"%>
<%@ page buffer="16kb"   autoFlash="false" %>          //缓冲区 16KB,溢出时抛出异常
<%@ page errorPage="error.jsp"%>                       //定义当前未捕获的异常处理页面
```

7.3.4 taglib 指令

1. taglib 指令的功能

taglib 指令用来定义一个标记库以及标记的前缀。

2. taglib 指令的语法规则

<%@ taglib uri="URIToTagLibrary" prefix="tagPrefix"%>

taglib 指令声明该 JSP 文件使用了自定义标记,并引用标记库,也指定了库中标记的前缀。不能使用 jsp、jspx、java、javax、servlet、sun 和 sunw 做前缀,这些前缀已经被 sun 公司声明保留。

7.3.5 JSP 指令应用案例

例 7.6 使用 page 指令的 session 属性,在页面上输出一个 sessionID。

ex7-06.jsp 代码清单如下:

```
<%@ page contentType="text/html; charset=gb2312"%>
<%@ page session="true"%>
<html>
<head><title>page 指令应用</title>
</head>
<body><center>
  <font size=5 face="隶书" color=blue>获得的 sessionID 是<br><hr>
  <%=session.getId()%>
  </font></center>
</body></html>
```

ex7-06.jsp 在浏览器中的显示效果如图 7-5 所示。



图 7-5 page 指令的 session 属性应用

7.4 JSP 动 作

7.4.1 JSP 动作功能

JSP 动作用来控制 JSP 引擎的行为，执行一些标准常用的 JSP 页面的动作，例如动态插入文件、重用 JavaBean 控件、设置 JavaBean 的属性、导向另一个页面、为 Java 插件(Plugin)生成 HTML 代码等。JSP 动作包含以下内容。

jsp:include：在页面运行时动态插入一个文件。

jsp:useBean：使用 JavaBean 控件。

jsp:setProperty：设置 JavaBean 属性。

jsp:getProperty：把 JavaBean 的属性插入到输出中。

jsp:forward：引导请求者进入新的页面。

jsp:plugin：插入一个 applet 或 Bean。

7.4.2 jsp:include 动作

1. jsp:include 动作功能

jsp:include 动作在即将生成的页面上动态地插入文件，它在页面运行时才将文件插入，对被插入文件进行处理。jsp:include 动作与 include 指令是有区别的，jsp:include 动作是动态的，而 include 指令是静态的。jsp:include 动作插入文件时，JSP 引擎不把插入文件和原 JSP 文件合并成一个新的 JSP 文件，而是在运行时把被插入文件包含进来。所以 JSP 页面与被插入的文件在逻辑和语法上是独立的。如果被插入文件被改动了，jsp:include 动作可以判断出来，并对被插入文件重新编译。如果被插入的文件是文本文件，直接把文件内容发送到客户端浏览器显示；如果被插入的是 JSP 文件，JSP 引擎执行该文件，然后把执行结果送客户端浏览器显示。而 include 指令是静态的，它把被嵌入文件插到当前位置后再进行编译。

jsp:include 动作中被引用的文件不能包含某些 JSP 代码，例如不能设置 HTTP 头等。通过包含 JSP 代码，也预包含了需要链接的 JSP 页面，使得应用开发比较灵活。

2. jsp:include 动作的语法规则

```
<jsp:include page="文件的 URL"/>
```

或

```
<jsp:include page="文件的 URL">
  <jsp:param name="参数名 1"  value="参数值 1"/>
  <jsp:param name="参数名 2"  value="参数值 2"/>
```

```
…
</jsp:include>
```

参数说明：

（1）page="文件的 URL"

设置需要插入文件的 URL，该参数是一个相对路径或代表相对路径的表达式。

（2）<jsp:param>

<jsp:param>子句可以把一个或多个参数传送到要插入的文件中去，一个页面可以使用多个<jsp:param>传递多个参数。传递参数时，被插入文件用以下语句获取传入的参数：

```
request.getParameter("参数名")
```

3. 应用举例

例 7.7 网上书店的新书展示栏目，可以为每一本新书制作一个文件，每个文件是一本新书介绍，在 ex7-07.jsp 代码中插入了 4 个文件。如果新书书目改变，只需要改变相关文件中的内容就可以了，而 ex7-07.jsp 页面可以不做改动。ex7-07.jsp 代码清单如下：

```
<%@ page language="java" %>
<%@ page contentType="text/html; charset=GB2312"%>
<html>
<head><title>jsp:include 动作</title>
</head>
<body>
<div align="center"><font size="6" face="隶书" color=blue><b>
新书展示<br></font>
    <table border="3">
        <tr>
           <td ><jsp:include page="newbook1.html"/></td>
           <td><jsp:include page="newbook2.html"/></td>
        </tr>
        <tr>
           <td><jsp:include page="newbook3.html"/></td>
           <td><jsp:include page="newbook4.html"/></td>
        </tr>
    </table></font>
</div>
</body></html>
```

其中一本新书的内容介绍文件 newbook1.html 代码清单如下：

```
<html><head><title>Windows 驱动开发技术详解</title>
</head>
<body>
  <table width="100%" border="0" >
```

```
        <tr>
            <td width="40%"rowspan="8"><img src="20302264.jpg"height="106"width="75"></td>
            <td colspan="1"><font size="1">Windows 驱动开发技术详解</font></td>
        </tr>
        <tr>
            <td colspan="1"><font size="1">张帆　等</font></td>
        </tr>
        <tr>
            <td colspan="1"><font size="1">电子工业出版社</font></td>
        </tr>
        <tr>
            <td colspan="1"><font size="1">2008-07-01</font></td>
        </tr>
        <tr>
            <td colspan="1"><font size="1">9787121068461</font></td>
        </tr>
        <tr>
            <td><font size="1">定价：¥65.00</font></td>
        </tr>
    </table></font>
</body></html>
```

文件 ex7-07.jsp 在浏览器中的显示效果如图 7-6 所示。



图 7-6　jsp:include 动作应用

例 7.8　ex7-08.jsp 文件使用<jsp:include>动作插入求阶乘的文件 ex7-08_1.jsp，并把求阶乘所需要的参数传递给 ex7-08_1.jsp 文件。

ex7-08.jsp 文件代码清单如下：

```
<%@ page contentType="text/html; charset=GB2312"%>
```

```
<html>
<head><title>jsp:include 动作</title>
</head>
<body>
<div align="center"><font size="5" face="隶书" color=blue><b>
    求一个数的阶乘<br></font>
    <jsp:include page="ex7-08_1.jsp">
          <jsp:param name="num" value="8"/>
    </jsp:include>
</div>
</body></html>
```

ex7-08_1.jsp 文件代码清单如下：

```
<%@ page contentType="text/html; charset=GB2312"%>
<html>
<head><title>jsp:include 动作</title>
</head>
<body>
  <% String str=request.getParameter("num");
      if(str==null){
        str="1";
      }
      int n=Integer.parseInt(str);
      double s=1;
      for(int i=1;i<=n;i++)
         s*=i;
   %>
  <div align="center"><font size="5" face="隶书" color=green><b>
    <%=n%>!=<%=s%></font>
  </div>
</body></html>
```

运行 ex7-08.jsp 文件，效果如图 7-7 所示。

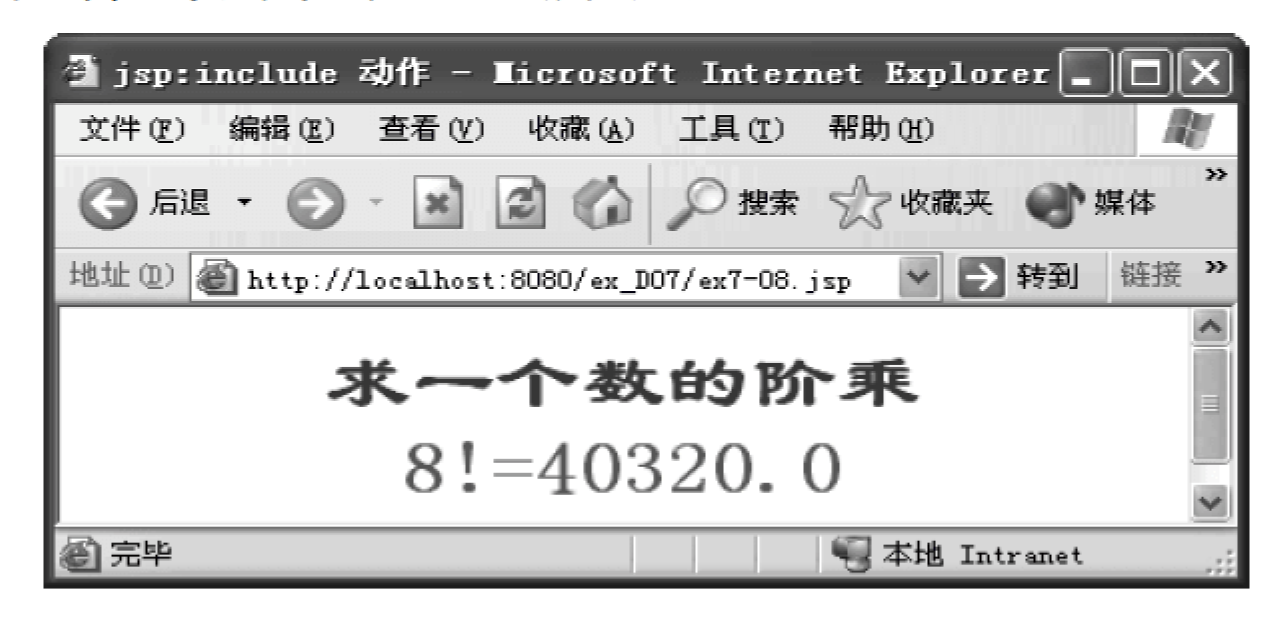


图 7-7　param 动作传递参数

7.4.3 jsp:forward 动作

1. jsp:forward 动作功能

jsp:forward 动作的功能是从一个 JSP 文件转向另一个文件。forward 动作停止当前页面的执行，转向另一个 HTML 或 JSP 页面。如果 JSP 页面包含了 jsp:forward 动作，则页面中的所有数据都不会发送到客户端，JSP 引擎也不再处理当前页面剩下的内容，缓冲区被清空。在客户端看到的是原页面的地址，而实际显示的是另一个页面的内容。forward 动作在控制型的 JSP 页面中经常使用。

2. jsp:forward 动作的语法规则

```
<jsp:forward page="要转向页面的 URL"/>
```

或

```
<jsp:forward page="要转向页面的 URL">
    <jsp:param name="参数名 1" value="参数值 1"/>
    <jsp:param name="参数名 2" value="参数值 2"/>
    …
</jsp:forward >
```

jsp:forward 动作只有一个 page 属性，指定目标文件的相对 URL，page 属性值可以是静态值，也可由 JSP 动态产生。

属性说明如下：

(1) page="要转向页面的 URL "

用来指明要转向的文件或 URL。这个文件可以是 JSP 程序段，也可以是其他能够处理 request 对象的文件(如 asp、cgi 和 php 等)。

(2) <jsp:param name="参数名" value="参数值">

向一个动态文件发送一个或多个参数及参数的值。name 指定参数名，value 设置参数值。如果要传递多个参数，可以在一个 JSP 文件中使用多个<jsp:param>标记。注意：如果使用了<jsp:param>标记，目标文件一定要是动态文件，这样才能够处理参数。

3. 应用举例

例 7.9 将文件 ex7-09.jsp 页面重新导向到 ex7-09_1.jsp 页面。

文件 ex7-09.jsp 代码清单如下：

```
<%@ page contentType="text/html; charset=GB2312"%>
<html>
<head><title>jsp:forward 动作</title>
</head>
<body>
```

```
      包含 jsp:forward 动作的 JSP 文件,文件中的数据不被输出。
   <jsp:forward page="ex7-09_1.jsp"/>
      数据不被输出。
</div>
</body></html>
```

文件 ex7-09_1.jsp 代码清单如下:

```
<%@ page contentType="text/html; charset=GB2312"%>
<html>
<head><title>jsp:forward 动作</title>
</head>
<body>
   <div align="center"><font size="5" face="隶书" color=blue><b>
      页面重新定向<br></font>
         <% out.println("forward 动作重新导向的第二个页面");%>
   </div>
</body></html>
```

ex7-09.jsp 代码在浏览器中的运行结果如图 7-8 所示。

读者可以注意到在文件 ex7-09.jsp 中的两行文字没有在浏览器中显示,浏览器地址栏目中显示的是文件 ex7-09.jsp 的 URL,而页面显示的是 ex7-09_1.jsp 文件的内容。

例 7.10 jsp:forward 动作在控制型的页面中特别有用。本例的任务是:输入图书信息后,可以有两个选择,如果选择"详细信息"单选钮,单击"确定"按钮后页面转向详细信息页面;如果选择"图书购买"单选钮,则转到图书购买页面。它由四个程序组成。

ex7-10.html:图书检索界面见图 7-9,读者在界面中可以选择重新定向的不同页面。

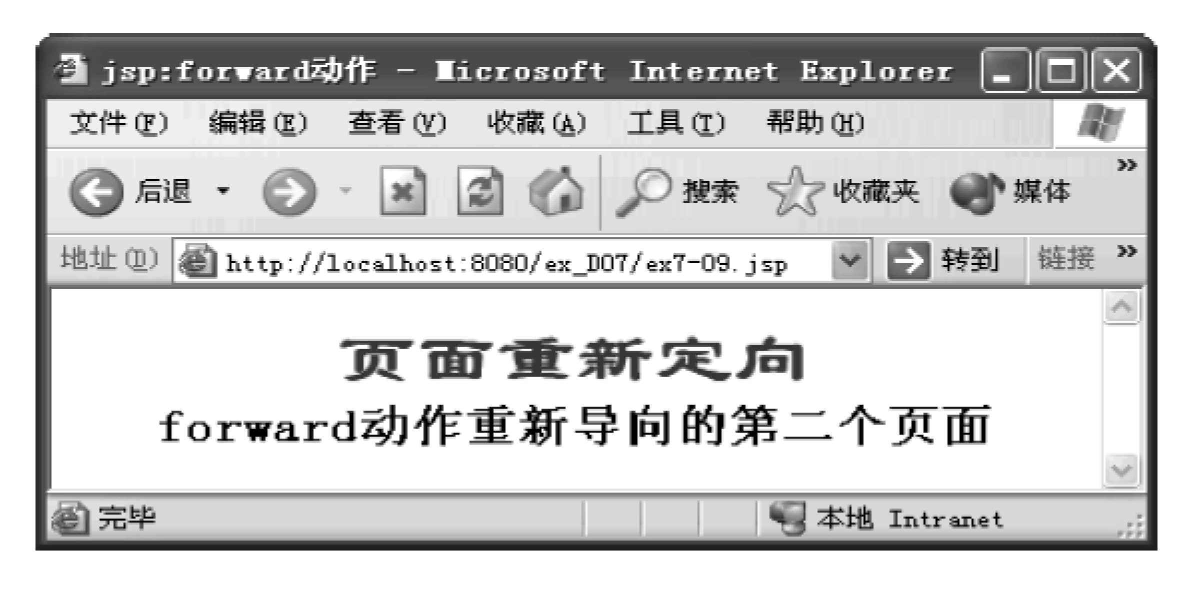


图 7-8 页面重新定向

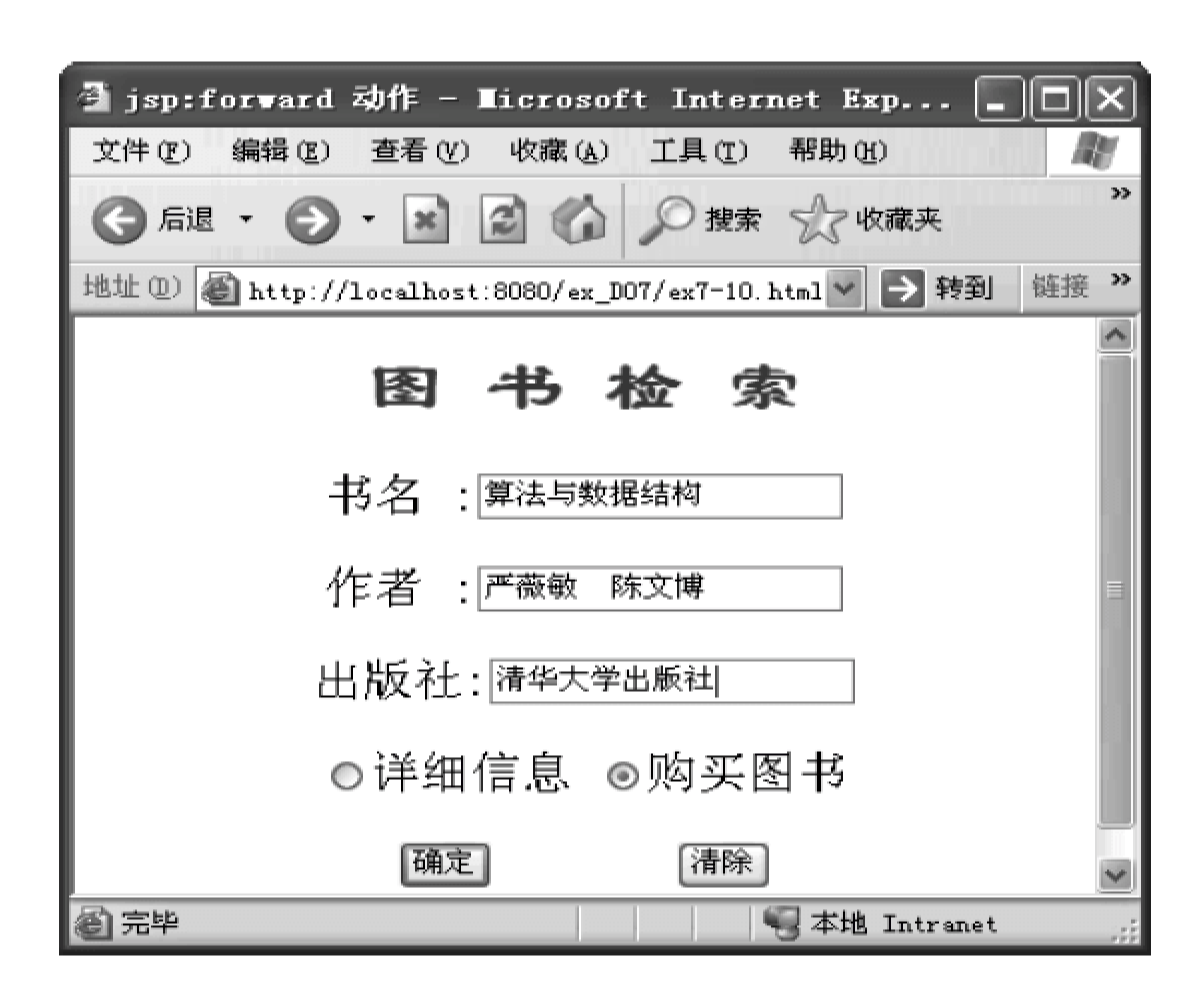


图 7-9 图书检索界面

ex7-10_1.jsp:判断读者选择的页面,根据读者的选择确定目标页面。

ex7-10_2.jsp 和 ex7-10_3.jsp:目标页面。

为使程序简单易读,对实际的源代码做了部分简化。

图书检索界面 ex7-10.html 将表单上读者选择的信息传递给 ex7-10_1.jsp。

ex7-10.html 源代码清单如下：

```
<html>
<head><title>jsp:forward 动作</title>
</head>
<body>
<center><b><font size="5" face="隶书" color=blue>图书检索</font></b><p>
<form method="post" action="ex7-10_1.jsp">
    <font size="3" face="楷体">
    书名  :<input type="text" name="bookname" size="20"><p>
    作者  :<input type="text" name="authname" size="20"><p>
    出版社:<input type="text" name="pubname" size="20"><br><p>
    <input type="radio" name="select" value="book" checked>详细信息
    <input type="radio" name="select" value="statistic">购买图书<p>
    <input type="submit" value="确定" name="B1">
    <input type="reset" value="清除" name="B2"></font>
    </center>
</form>
</body></html>
```

ex7-10_1.jsp 页面判断读者的选择，接受 ex7-10.html 发来的信息，根据读者在单选钮中的选择，转向目标页。

ex7-10_1.jsp 代码清单如下：

```
<%@ page contentType="text/html; charset=GB2312"%>
<% request.setCharacterEncoding("GB2312"); %>
<html>
<head><title>jsp:forward 动作</title></head>
<body>
<%
  String getBook,getAuth,getPub,getselect;
  getBook=request.getParameter("bookname");
    //使用 request 对象的 getParameter 方法获取 ex7-10.html 页面的参数
  getAuth=request.getParameter("authname");
  getPub=request.getParameter("pubname");
  getselect=request.getParameter("select");
  if(getselect.equals("book")){
    //如果用户选择"详细信息",转向 ex7-10_2.jsp 页面
%>
    <jsp:forward page="ex7-10_2.jsp">
    <jsp:param name="bookname" value="<%=getBook%>"/>
    <jsp:param name="authname" value="<%=getAuth%>"/>
    <jsp:param name="pubname" value="<%=getPub%>"/>
    </jsp:forward>
```

```
<%
  }else{
  //选择"购买图书",转向 ex7-10_3.jsp 页面
%>
    <jsp:forward page="ex7-10_3.jsp">
    <jsp:param name="bookname" value="<%=getBook%>"/>
    <jsp:param name="authname" value="<%=getAuth%>"/>
    <jsp:param name="pubname" value="<%=getPub%>"/>
    </jsp:forward>
<%
  }
%>
</body></html>
```

ex7-10_2.jsp 显示图书详细信息的页面代码清单如下：

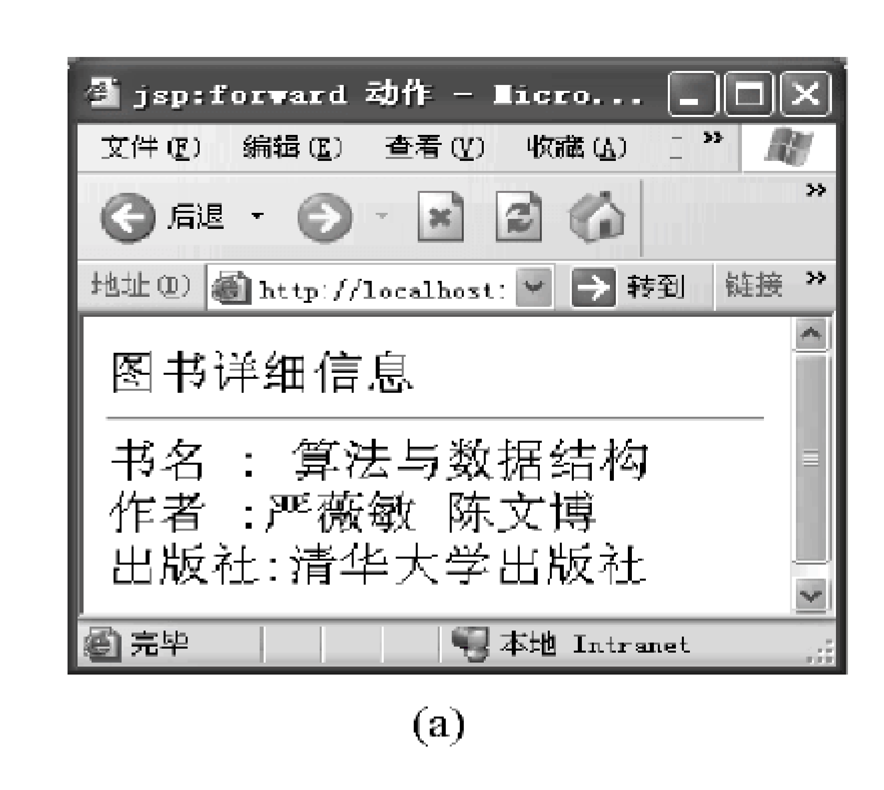
```
<%@ page contentType="text/html; charset=GB2312"%>
<% request.setCharacterEncoding("GB2312");%>
<html>
<head><title>jsp:forward 动作</title></head>
<body>图书详细信息
<%
  String getBook,getAuth,getPub;
  getBook=request.getParameter("bookname");
  getAuth=request.getParameter("authname");
  getPub=request.getParameter("pubname");
%>
<hr>
  书名  : <%=getBook%><br>
  作者  :<%=getAuth%><br>
  出版社:<%=getPub%>
</body></html>
```

ex7-10_3.jsp 显示购买图书页面代码清单如下：

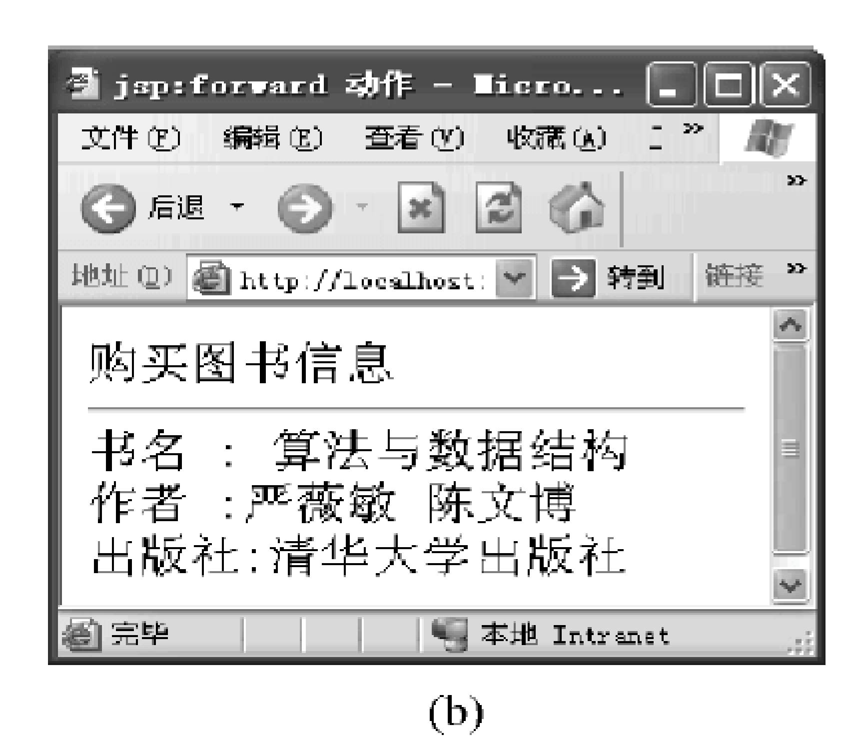
```
<%@ page contentType="text/html; charset=GB2312"%>
<% request.setCharacterEncoding("GB2312"); %>
<html>
<head><title>jsp:forward 动作</title></head>
<body>购买图书信息
<%
  String getBook,getAuth,getPub;
  getBook=request.getParameter("bookname");
  getAuth=request.getParameter("authname");
  getPub=request.getParameter("pubname");
%>
```

```
<hr>
  书名  :<%=getBook%><br>
  作者  :<%=getAuth%><br>
  出版社:<%=getPub%><br>
</body></html>
```

在浏览器中的显示效果见图 7-10。



(a)



(b)

图 7-10 根据用户选择页面定向

7.4.4 jsp:plugin 动作

1. jsp:plugin 动作功能

jsp:plugin 动作的功能是将服务器端的 Java 小应用程序(Applet)或 JavaBean 组件下载到浏览器端去执行,相当于在客户端浏览器插入 Java 插件。

当 JSP 文件被编译,传输到浏览器时,<jsp:plugin>动作将动态产生<object>或<embed>元素标记,以使浏览器的 Java 插件运行 Applet。

一般来说 jsp:plugin 动作会指定对象是 applet 还是 Bean,同时也会指定字节码文件(.class)的名称与位置,另外还会指定从哪里下载 Java 插件。

2. jsp:plugin 动作的语法规则

```
<jsp:plugin  type="bean|applet"  code="保存类的文件名"  codebase="类路径"
    [name="对象名"]
    [archive="相关文件路径"]
    [align="bottom|top|middle|left|right"]          //对齐方式
    [height="displayPixels"]                        //高度
    [width="displayPixels"]                         //宽度
    [hspace="leftRightPixels"]                      //水平间距
    [vspace="topBottomPixels"]                      //垂直间距
    [jreversion="Java 环境版本"]
    [nspluginurl="供 NC 使用的 plugin 加载位置"]
    [iepluginurl="供 IE 使用的 plugin 加载位置"]>
```

```
<jsp:params>
    <jsp:param name="参数名 1"  value="参数值 1"/>
    <jsp:param name="参数名 2"  value="参数值 2"/>
    ...
</jsp:params>
[<jsp:fallback>错误信息</jsp:fallback>]
</jsp:plugin>
```

plugin 指令属性说明如下：

(1) type="bean|applet"

指定将被执行的插件对象的类型是 Bean 还是 Applet。因为这个属性没有默认值，所以必须指定一个。

(2) code="保存类的文件名"

指定 Java 插件将要执行的字节码(Java Class)文件的名字，其后缀必须是.class。这个文件必须保存在由 codebase 属性指定的目录里。

(3) codebase="类路径"

说明将要被下载的 Java Class 文件的目录(或是路径)，如果没有提供该项属性，那么默认为使用<jsp:plugin>动作的 JSP 文件的路径。

(4) name="对象名"

bean 或 applet 实例的名字。

(5) archive="相关文件路径"

一些由逗号分开的路径名，预装一些将要使用的 class，用来提高 applet 的性能。

(6) Align="bottom|top|middle|left|right"

将要显示的图形、对象和 Applet 的位置，可取以下值。

bottom：位于底部。

top：位于顶部。

middle：位于中间。

left：位于左边。

right：位于右边。

(7) height="displayPixels" width="displayPixels"

将要显示的 Applet 或 Bean 的长与宽，以像素为单位。

(8) hspace="leftrightPixels" vspace="topbottomPixels"

Applet 或 Bean 显示时在屏幕左、右、上、下需要留下的空间，以像素为单位。

(9) jreversion="Java 环境版本"

运行 Applet 或 Bean 所需要的 Java Runtime Environment(JRE)的版本，默认值是 1.1。

(10) nspluginurl="供 NC 使用的 plugin 加载位置"

Netscape Navigator 能够使用的 JRE 的下载地址，它是一个标准的 URL。

(11) iepluginurl="供 IE 使用的 plugin 加载位置"

IE 用户能够使用的 JRE 的下载地址，它也是一个标准的 URL。

(12) <jsp:params>

需要向 applet 或 Bean 传送的参数或参数值。

(13) <jsp:fallback>错误信息</jsp:fallback>

一段文字，当 Java 插件不能启动时，这段文字向用户显示；如果插件能够启动而 applet 或 Bean 不能执行，那么浏览器弹出一个错误信息。

3. 应用举例

例 7.11 使用 jsp:plugin 动作下载名为 RollingMessage.java 的 Java 小程序，并在 applet 中输出一行滚动显示的文字"欢迎学习"Web 技术应用基础"!"。

ex7-11.jsp 代码清单如下：

```
<%@ page contentType="text/html; charset=GB2312"%>
<html>
<head><title>用 &ltjsp:plugin&gt 加载 Applet</title>
</head>
<body><center>
<font size=3 face="隶书" color=blue>用 &ltjsp:plugin&gt 加载 Applet</font>
  <! -- 用 plugin 加载 Applet -->
    <jsp:plugin type="applet" code="RollingMessage.class" height="60" width="550" >
    </jsp:plugin>
</center>
</body></html>
```

RollingMessage.java 程序要先编译，形成字节码文件 RollingMessage.class，与 ex7-11.jsp 文件存放在同一目录中。

RollingMessage.java 程序清单如下：

```
import java.awt.*;
import java.awt.Color;
import java.awt.Font;
import java.awt.Graphics;
public class RollingMessage extends java.applet.Applet implements Runnable{
  Thread runThread;
  String s="欢迎学习"Web 技术应用基础"!";
  int s_length=s.length();
  int x_character=0;
  Font wordFont=new Font("楷体_GB2312" , Font.BOLD ,30);
  public void start(){
     if(runThread==null){
        runThread=new Thread(this);
        runThread.start();
     }
  }
  public void stop(){
     if(runThread!=null){
        runThread.stop();
```

```
            runThread=null;
        }
    }
    public void run(){
        while(true){
            if(x_character++>s_length)
        x_character=0;
            repaint();
            try{
                Thread.sleep(300);
            }
            catch (InterruptedException e){}
        }
    }
    public void paint(Graphics g) {
        g.setFont (wordFont);
        g.setColor (Color.blue);
        g.drawString (s.substring(0,x_character), 8, 50);
    }
    public static void main(String args[]) {          //Application 程序入口
        Frame f=new Frame("动画程序");                  //创建 Application 程序框架
        RollingMessage drawTest=new RollingMessage();
        drawTest.init();
        drawTest.start();
        f.add("Center",drawTest);
        f.resize(400, 100);
        f.show();
    }
}
```

ex7-11.jsp 代码运行结果如图 7-11 所示,图中文字"欢迎学习"Web 技术应用基础"!"水平滚动。



图 7-11　jsp:plugin 动作应用

7.5 jsp:useBean 动作

使用 jsp:useBean 动作定义 JSP 页面要使用的 JavaBean 对象，与 useBean 动作配合使用的有：jsp:setProperty 和 jsp:getProperty 动作。使用 jsp:setProperty 动作在 JSP 页面中设置 JavaBean 对象的属性；而用 jsp:getProperty 动作将 JavaBean 对象属性值转化为一个字符串，放入内置输出对象，并把它输出显示。

7.5.1 jsp:useBean 动作功能

Java 的 bean 是一种可以重复使用的软件组件。JavaBean 是 Java 类的一种，通过封装属性和方法成为具有某种功能或处理某种业务的对象，简称 bean。一个 Web 应用往往包括：数据层、业务逻辑层和表示层。一个基本的 JSP 页面包含了 HTML 标记和 Java 程序段，将表示层和业务逻辑层混杂在一起，使得页面显得混乱，不好维护。应用 JavaBean 将表示层和业务逻辑层分开，把业务的逻辑处理过程交给 bean 完成，再由 JSP 页面去调用 bean。bean 降低了 JSP 程序的复杂度，同时也增加了软件的可重用性。

jsp:useBean 动作用来装载一个将要在 JSP 页面中使用的 JavaBean。它创建一个 bean 实例并指定其名字和作用范围。

读者可以把 JavaBean 理解为一个黑箱，只要知道它的功能和接口，就可使用。JavaBean 有三种接口，可以独立开发，具体如下：

（1）JavaBean 可以调用的方法。

（2）JavaBean 提供的可以读写的属性。

（3）JavaBean 向外部发送或从外部接收的事件。

7.5.2 jsp:useBean 语法规则

jsp:useBean 语法格式如下：

```
<jsp:useBean id="beanInstanceName" class="packsge.class"
    scope="page|request|session|application"/>
</jsp:useBean>
```

属性说明：

（1）id="beanInstanceName"

定义 bean 组件的名字，使它能够在以后的程序中使用，并使用 bean 组件的名字来分辨不同的 bean。该变量名区分大小写，必须符合所使用的脚本语言的规定。如果这个 bean 已经在“<jsp:useBean>”标记中创建，这个 id 的值必须与原来的 id 值一致。

（2）scope="page|request|session|application"

该属性确定了 JavaBean 的使用范围，或称生命周期。JavaBean 只有在它定义的范围

内有效。默认值是 page,以下是它的详细说明。

① page:JSP 引擎为每个用户分配不同的 bean,尽管每个用户的 bean 功能一样,但是它们占有不同的内存空间。该 bean 只在当前页面有效,当用户离开该页面时,JSP 引擎取消分配给该用户的 bean。

② request:该 bean 的有效范围是 request 期间。在任何执行相同请求的 JSP 文件中使用该 bean,直到页面执行完毕向客户端发回响应或转到另一个文件为止。可以使用 request 对象访问该 bean,例如 request. getAttributes(beanInstanceName)。

③ session:该 bean 的有效范围是会话期间。从创建 bean 开始,就能在任何使用相同 bean 的 JSP 文件中使用该 bean。如果在一个 session 期,用户访问了多个页面,这些页面都包含有一个 useBean 标记,这些 useBean 标记中的 id 值一样,那么,用户在这些页面中使用的 bean 是同一个 bean。也就是说,如果用户在某个页面中改变了 bean 的某个属性值,则其他页面的该 bean 的该属性值也发生相同的变化。在创建 bean 的 JSP 文件的"<%@page%>"指令中必须指定 session="true"。

④ application:从创建 bean 开始,就能在任何相同的 application 的 JSP 文件中使用该 bean,该 bean 存在于整个 application 生存周期内,任何共享此 application 的 JSP 文件都能使用同一 bean。即所有的用户共享一个 bean,如果有某个用户更改了该 bean 的某个属性值,那么所有用户的该 bean 的该属性值都发生相同的变化,该 bean 一直到服务器关闭时才被取消。

(3) class="package. class"

使用 new 关键字以及类构造函数从一个类中实例化一个 bean。这个 class 不能是抽象的,必须有一个公用的没有参数的构造函数。package 名字要区分大小写。

7.5.3 jsp:useBean 工作过程

当一个包含有 useBean 标记的 JSP 页面在服务器端加载运行时,JSP 引擎根据 useBean 中 id 属性指定的名字,在一个同步块中查找内置对象 pageContent 中是否包含该 id 指定的名字和 scope 指定的作用域的对象,如果该对象存在,JSP 引擎把这样一个对象分配给用户,用户则获得一个 id 属性指定名字、scope 属性指定作用域的 bean。

7.5.4 jsp:useBean 应用实例

例 7.12 本例说明了 jsp:useBean 具有不同的 scope 属性值的应用。

1. 任务要求

制作一个简单的 bean,用 Java 编写的 Box 类,可以设置长方体的长、宽、高,并求长方体的表面积和体积。在 JSP 页面中使用 bean,并显示不同 scope 属性值 JavaBean 的使用范围。

2. 字节码文件的存放目录

在 Web 服务目录下建立字节码文件的存放目录 WEB-INF\classes，由于本章例题存放在 D:\Tomcat 5.5\webapps\ex_D07 目录下，所以本章字节码的存放目录是 D:\Tomcat 5.5\webapps\ex_D07\WEB-INF\classes\bean。用 Java 编写的立方体类 Box.java 编译生成的字节码文件 Box.class 存放在该目录下。

3. Box.java

Box.java 代码清单如下：

```
package bean;
import java.io.*;
public class Box{
  int length,width,height;
  public Box(){
    length=1;
    width=1;
    height=1;
  }
  public void setlength(int newlength){
    length=newlength;
  }
  public void setwidth(int newwidth){
    width=newwidth;
  }
  public void setheight(int newheight){
    height=newheight;
  }
  public int getlength(){
    return length;
  }
  public int getwidth(){
    return width;
  }
  public int getheight(){
    return height;
  }
  public int BoxVolume(){
    return length * width * height;
  }
  public double BoxArea(){
    return 2 * (length * width+width * height+length * height);
  }
```

```
}
```

Box.java 编译通过后生成字节码文件 Box.class，将该字节码文件存放在 Tomcat 5.5\webapps\ex_D07\WEB-INF\classes\bean 目录下。

4. scope="page"，bean 的有效范围是当前页

在 ex7-12.jsp 页面中，获得一个 id 为 MyBox，作用域是 page 的 bean，该 bean 的生命期为当前页。ex7-12.jsp 代码清单如下：

```
<%@ page contentType="text/html; charset=GB2312"%>
<%@ page import="bean.Box"%>
<html>
<body>
    <jsp:useBean id="MyBox" scope="page" class="bean.Box">
    </jsp:useBean>
    <%MyBox.setlength(20);%>
    Box 的长是：
    <%=MyBox.getlength()%>
    <%MyBox.setwidth(10);%><br>
    Box 的宽是：
    <%=MyBox.getwidth()%>
    <%MyBox.setheight(5);%><br>
    Box 的高是：
    <%=MyBox.getheight()%><br>
    Box 的体积是：
    <%=MyBox.BoxVolume()%><br>
    Box 的表面积是：
    <%=MyBox.BoxArea()%>
</body></html>
```

图 7-12　作用域为 page 的 beanJSP 页面

在浏览器中的显示效果如图 7-12 所示。

5. scope="session"，用户在不同页面共享 bean

一个用户可以在不同页面共享一个 bean，其作用域是 session 的 bean。本案例设计是：用户在 ex7-12_1.jsp 和 ex7-12_2.jsp 页面间共享同一个立方体 bean，如果用户在其中一个页面中更新了长方体的数据，那么刷新另一个页面，则可见到更新后的数据。

(1) ex7-12_1.jsp 代码清单

```
<%@ page contentType="text/html; charset=GB2312"%>
<%@ page import="bean.Box"%>
<html><body>
    <jsp:useBean id="MyBox"  class="bean.Box" scope="session">
    </jsp:useBean>
    Box 的长是：
```

```
    <%=MyBox.getlength()%><br>
    Box 的宽是:
    <%=MyBox.getwidth()%><br>
    Box 的高是:
    <%=MyBox.getheight()%><br>
    Box 的体积是:
    <%=MyBox.BoxVolume()%><br>
    Box 的表面积是:
    <%=MyBox.BoxArea()%><br>
    <a href=ex7-12_2.jsp>ex7-12_2.jsp</a>
</body></html>
```

(2) ex7-12_2.jsp 代码清单

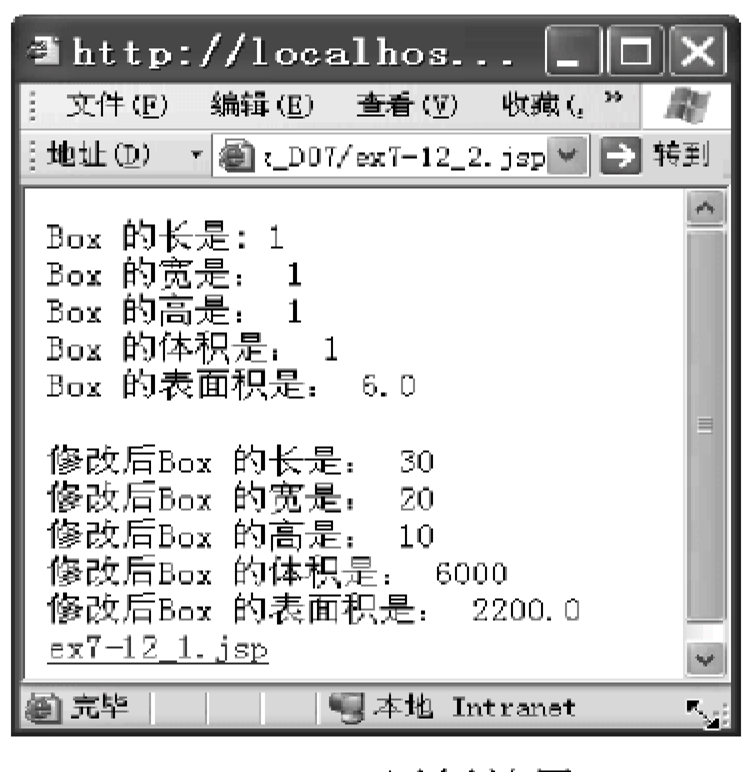
```
<%@ page contentType="text/html; charset=GB2312"%>
<%@ page import="bean.Box"%>
<html>
<body>
    <jsp:useBean id="MyBox" scope="session" class="bean.Box">
    </jsp:useBean>
    Box 的长是:
    <%=MyBox.getlength()%><br>
    Box 的宽是:
    <%=MyBox.getwidth()%><br>
    Box 的高是:
    <%=MyBox.getheight()%><br>
    Box 的体积是:
    <%=MyBox.BoxVolume()%><br>
    Box 的表面积是:
    <%=MyBox.BoxArea()%><p>
    <%MyBox.setlength(30);%>
    修改后 Box 的长是:
    <%=MyBox.getlength()%>
    <%MyBox.setwidth(20);%><br>
    修改后 Box 的宽是:
    <%=MyBox.getwidth()%>
    <%MyBox.setheight(10);%><br>
    修改后 Box 的高是:
    <%=MyBox.getheight()%><br>
    修改后 Box 的体积是:
    <%=MyBox.BoxVolume()%><br>
    修改后 Box 的表面积是:
    <%=MyBox.BoxArea()%><br>
    <a href=ex7-12_1.jsp>ex7-12_1.jsp</a>
</body></html>
```

（3）代码在浏览器中的显示效果

ex7-12_1.jsp 在浏览器中的显示效果如图 7-13(a)所示。单击此图中的超链接，页面转跳至 ex7-12_2.jsp，显示长方体的数据并将其数值更新，显示效果如图 7-13(b)所示，单击此图中的超链接，页面又跳回 ex7-12_1.jsp 页面，但长方体的数值已被更新。说明用户在 ex7-12_1.jsp 和 ex7-12_2.jsp 页面间共享一个 bean。



(a) 7-12_1.jsp 运行效果



(b) 7-12_2.jsp 运行效果



(c) 再次运行 7-12_1.jsp

图 7-13　页面共享 beanex

6. scope="application"，不同用户共享 bean

在 ex7-12_3.jsp 代码中创建了一个 id 为 MyBox，作用域是 application 的 bean。该 bean 由所有用户共享，直到服务器被关闭为止。ex7-12_3.jsp 代码清单如下：

```
<%@ page contentType="text/html; charset=GB2312"%>
<%@ page import="bean.Box"%>
<html><body>
   <jsp:useBean id="MyBox" scope="application" class="bean.Box">
   </jsp:useBean>
   Box 的长是：
   <%=MyBox.getlength()%><br>
   <%MyBox.setlength(40);%>
   修改后 Box 的长是：
   <%=MyBox.getlength()%><p>

   Box 的宽是：
   <%=MyBox.getwidth()%>
   <%MyBox.setwidth(30);%><br>
   修改后 Box 的宽是：
   <%=MyBox.getwidth()%><p>

   Box 的高是：
   <%=MyBox.getheight()%>
```

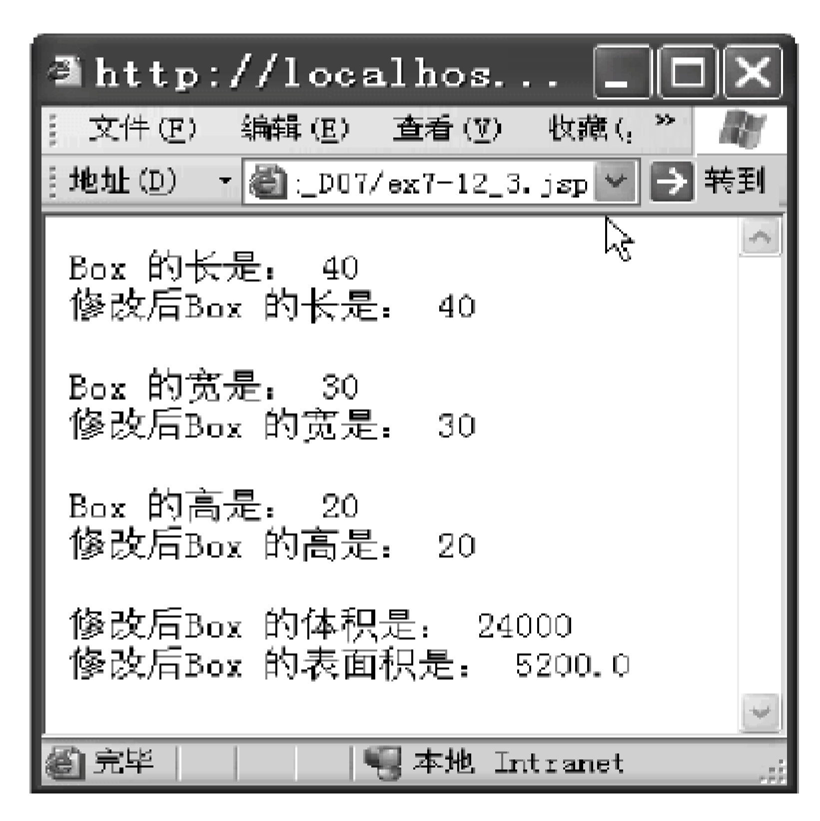
```
    <% MyBox.setheight(20);%><br>
    修改后 Box 的高是：
    <%=MyBox.getheight()%><p>

    修改后 Box 的体积是：
    <%=MyBox.BoxVolume()%><br>
    修改后 Box 的表面积是：
    <%=MyBox.BoxArea()%>
</body></html>
```

第一个用户运行 7-12_3.jsp 显示长方体的数据，然后更新这些数据并显示，效果如图 7-14(a)所示。其他后续用户运行 7-12_3.jsp 显示长方体更新后的数据，如图 7-14(b)所示。




(a) 第一个用户访问 7-12_3.jsp



(b) 第二个用户访问 7-12_3.jsp

图 7-14　不同用户共享 bean

7. 创建类实例应用 bean

例 7.13　本例(ex7-13.jsp)的运行效果与 ex7-12.jsp 的运行效果是一样的。

在 ex7-12.jsp 页面中，使用以下语句创建名为 MyBox 的 bean：

```
<jsp:useBean id="MyBox" scope="page" class="bean.Box" >
</jsp:useBean>
```

在 ex7-13.jsp 代码中使用语句：<%Box MyBox=new Box();%>，创建名为 MyBox 的 bean。

ex7-13.jsp 代码清单如下：

```
<%@ page contentType="text/html; charset=GB2312"%>
<%@ page import="bean.*"%>
<html><body>
    <%Box MyBox=new Box();%>
    <%MyBox.setlength(20);%>
```

```
    Box 的长是：
    <%=MyBox.getlength()%>
    <%MyBox.setwidth(10);%><br>
    Box 的宽是：
    <%=MyBox.getwidth()%>
    <%MyBox.setheight(5);%><br>
    Box 的高是：
    <%=MyBox.getheight()%><br>
    Box 的体积是：
    <%=MyBox.BoxVolume()%><br>
    Box 的表面积是：
    <%=MyBox.BoxArea()%>
</body></html>
```

7.5.5 设置和获取 bean 属性值

1. jsp:setProperty 动作

(1) jsp:setProperty 动作的功能

用来设置 bean 中的属性值。

(2) jsp:setProperty 动作的语法规则

<jsp:setProperty name="bean 的名字"property="bean 的属性"value="<%=expression%>"/>

(3) 属性说明

① name="bean 的名字"：表示已经在"<jsp:useBean>"中创建的 bean 实例名字。

② property="bean 的属性"：用于匹配 bean 中的属性。

③ value="<%=expression%>"：存储用户在 JSP 输入的属性值。

(4) 使用说明

<jsp:setProperty>使用 bean 给定的 set XXX()方法，在 bean 中设置一个或多个属性值，在使用该动作前必须使用<jsp:useBean>声明此 bean。<jsp:useBean>和<jsp:setProperty>是相互关联的，<jsp:setProperty>中的 name 的值应当和<jsp:useBean>中的 id 的值相同。

使用"<jsp:setProperty>"来设定属性值时，可以使用多种不同方法，例如：

① 通过用户输入的所有值(作为参数存储在 request 对象中)来匹配 bean 中的属性。

② 通过用户输入的指定值来匹配 bean 中指定的属性。

③ 在运行时使用一个表达式来匹配 bean 的属性。

2. jsp:getProperty 动作

(1) jsp:getProperty 动作的功能

使用 jsp:getProperty 获取 bean 的属性值，并把它们在 JSP 页面中显示出来。

(2) jsp:getProperty 动作的语法规则

<jsp:getProperty name="bean 的名字" property="bean 的属性"/>

7.6 JSP 指令与动作的应用——读者选购图书

1. 功能介绍

例 7.14 读者选购图书的应用,其中包含了三个程序。

ex7-14.html: 是读者选择图书的界面,见图 7-15,读者在界面内选择需要的图书和数量,单击"确定"按钮后,将信息发送给 ex7-14.jsp。



图 7-15 读者选择图书

ex7-14.jsp: 接收 ex7-14.html 的信息,使用 jsp:useBean 动作创建了名为 book 的 bean 组件。ex7-14.jsp 使用 jsp:setProperty 动作在 bean 中设置属性值。

BookBean.java: 是一个 JavaBean,将它编译成字节码文件 BookBean.class 存放在 Tomcat 5.5\webapps\ex_D07\WEB-INF\classes\bean 目录下。

2. ex7-14.html 代码清单

```
<html><head><title>图书订购</title>
</head>
<body><center>
<font size="5" face="隶书" color="#0000ff">请选择图书</font>
</center><hr>
<form method="post" action="ex7-14.jsp">
  <div align="center"><font size=4>
      书名:
      <select name="bookName">
        <option value="C++语言程序设计" selected>C++语言程序设计</option>
        <option value="Google 搜索从入门到精通">Google 搜索从入门到精通</option>
        <option value="现代英语教程学习辅导(2)">现代英语教程学习辅导(2)</option>
```

```
        <option value="火星的故事">火星的故事</option>
        <option value="考研英语听力模拟试题">考研英语听力模拟试题</option>
        <option value="WTO 与中国基础教育发展">WTO 与中国基础教育发展</option>
    </select>
    数量：
    <select name="bookNum">
        <option value="1"selected>01</option>
        <option value="2">02</option>
        <option value="3">03</option>
        <option value="4">04</option>
        <option value="5">05</option>
        <option value="6">06</option>
    </select><p>
    <input type="submit" value="确定 " name="submit">
    <input type="reset" value="清除 " name="reset">
  </div>
</form>
</body></html>
```

3. ex7-14.jsp 代码清单

```
<%@ page contentType="text/html; charset=GB2312"%>
<% request.setCharacterEncoding("GB2312");%>
<%@ page import="bean.BookBean"%>
<jsp:useBean id="book" class="bean.BookBean" scope="request">
    <jsp:setProperty name="book" property="bookName"/>
    <jsp:setProperty name="book" property="bookNum"/>
</jsp:useBean>
<html><head><title>图书订购</title>
</head>
<body><center>
<font size="5" face="隶书" color="#0000ff">订购图书清单</font>
</center><br>
<div align="center"><font size="4">
  书名：<%=book.getBookName()%><BR>
  数量：<%=book.getBookNum()%><BR>
  </font>
</div>
</body></html>
```

4. BookBean.java 代码清单

```
package bean;
```

```
import java.io.*;
public class BookBean{
    private String BookName="";
    private int BookNum=1;
    public BookBean(){
    }
    public void setBookName( String BookName ){
this.BookName=BookName;
    }
    public String getBookName(){
return this.BookName;
    }
    public void setBookNum( int BookNum ){
this.BookNum=BookNum;
    }
    public int getBookNum(){
        return this.BookNum;
    }
}
```

5. 运行结果

在浏览器中运行 ex7-14.html，在表单中输入数据，单击“确定”按钮，信息处理后在浏览器中显示如图 7-16 所示。



图 7-16 运行结果

习题、上机练习与实训 7

一、习题

1. 简述 JSP 的主要特点。
2. JSP 文件名的后缀是什么？JSP 代码的定界符是什么？
3. 简述 JSP 的运行过程。
4. 一个 JSP 页面的源代码与在客户端看到的代码是否一致？

5. 举例说明 JSP 常用的 5 种语句类型。

6. 什么是 HTML 注释？什么是隐藏注释？它们分别在什么情况下使用？

7. HTML 注释在浏览器中能见到吗？在客户端的“查看源文件”中能见到吗？

8. 在什么标记之间声明变量和方法？使用时应注意些什么？

9. 在什么标记之间插入表达式？使用时应注意些什么？

10. JSP 指令的主要功能有哪些？

11. 如果 include 指令嵌入的是静态文件，被嵌入的文件在什么时间被编译？如果 include 指令包含的是动态文件，被包含的文件在什么时间被编译？

12. page 指令的作用是什么？并举出其 3 种属性的应用。

13. jsp:include 动作与 include 指令的区别是什么？

14. 应用 JavaBean 的好处是什么？

15. jsp:setProperty 动作的功能是什么？

16. jsp:getProperty 动作的功能是什么？与 jsp:setProperty 动作如何配合使用？

二、上机练习

1. 使用 JSP 技术在浏览器中输出四行由大变小的文字。

2. 使用 Date 函数读取系统当前时间，根据不同的时间段，在浏览器输出不同的问候语，例如上午 0～12 点之间输出“早上好”，同时把系统的时间年、月、日、小时、分、秒和星期输出到用户的浏览器。

3. 加载静态文件。制作一个 HTML 文件，在页面上输出一行文字，例如“include 指令的使用。”等。然后制作一个 JSP 文件，应用 include 指令加载刚制作的 HTML 文件，在客户端显示出来，并在客户端的“查看源文件”中观察源文件。

4. 加载动态文件，制作一个 JSP 文件，计算一个数的平方根，然后再制作一个 JSP 文件，应用<jsp:include>动作加载刚制作的 JSP 文件，在客户端显示出来，并在客户端的“查看源文件”中观察源文件。

5. 使用 jsp:include 动作制作一个班级的 JSP 页面，显示班上每个同学的信息。每个同学都有自己的页面，由本人定期更新。要求班级的 JSP 页面是稳定的，看谁的页面最有特色。

6. 设计表单，制作读者选购图书的界面，当读者选中一本图书后，单击“确定”按钮，页面跳转到介绍该图书信息页面，请用“jsp:forward page=”语句。

7. 制作一个页面，页面的背景颜色，文字的大小、字体和颜色可以根据用户的选择来实施。界面如图 7-17 所示，要求应用 JavaBean 完成。



图 7-17 改变色彩的界面

三、实训课题

1. 应用 JavaBean 完成一个计算器的制作。

2. 应用 JavaBean 完成一个计数器的制作,计数器 bean 的范围是 application 的,由不同用户共享。

3. 制作一个购物车 bean,要求具有选择商品、添加商品到购物车、查看购物车和从购物车中删除商品等功能。

第 8 章 JSP 内置对象

为了简化 JSP 的应用,JSP 提供了 9 个内置对象。这些内置对象可以直接引用,不需要显式声明,也不需要实例化。

JSP 内置对象案例应存放在 Web 服务目录下,本书存放在 D:\Tomcat 5.5\webapps\ex_D08 目录下。

8.1 JSP 内置对象概述

在 JSP 中预先定义好一些常用的对象,在 Web 应用中可以直接使用这些对象。内置对象的构建基础是 HTTP 协议,可以使用这些对象完成收集浏览器请求发出的信息、响应浏览器以及存储用户信息等工作,内置对象的应用大大简化了 Web 开发工作。

因为 Java 是区分大小写的,所以在 JSP 中对象名书写一定要准确,特别要注意字母的大小写。

JSP 常用内置对象及它们的作用见表 8-1。

表 8-1 JSP 内置对象

内置对象名称	作 用
request	客户端请求。获取用户提交的请求信息,例如在 form 表单中输入的信息。该请求包含来自 get/post 请求的参数
response	服务器对客户端的回应。使用该对象响应用户请求,发送回应信息给用户
session	与请求有关的会话对象。客户端与服务器端的一次会话,在客户端与服务器端断开之前,都可以访问 session 的有关属性和方法。session 对象可以存储用户的状态信息
application	与服务器环境相关的对象
out	向客户端发送信息的对象,用来传送回应的输出信息流
config	脚本程序配置对象。提供一些配置信息
pageContext	管理网页的上下文属性,代表的是当前页面运行的一些属性
page	代表正在运行的由 JSP 文件产生的类对象
exception	JSP 运行时产生的异常对象

8.2 request 对象

request 对象和 response 对象是 JSP 中应用较多的两个对象，request 对象用于得到用户提交的信息，而 response 对象是向用户发送响应信息，两者结合起来完成动态页面的交互功能。

8.2.1 request 对象的功能

当用户在客户端向 Web 服务器提出请求时，Web 服务器响应该请求，作出回应。为此，JSP 内置了 request 和 response 对象。request 对象主要用来获取用户信息，它包含了客户端的相关信息。response 对象主要用来向客户发送信息，它包含了 Web 服务器对客户的响应信息。

JSP 的内置对象 request 封装了用户提交的信息，使用 request 对象调用相应的方法可以获得所需要的封装信息。例如，使用表单向 Web 服务器提交信息：

```
<form methed=get|post action="服务器端应用程序 URL">
```

用 post 或 get 方法提交 HTML 表单信息。应用 post 方法提交的数据不被显示，比较安全。而使用 get 方法，提交的数据会附加到请求的 URL 之后，将在地址栏目中显示出来，存在不安全因素，例如，在表单中向服务器端的 plugin.jsp 文件提交表单信息，表单信息将附在 plugin.jsp 文件的 URL 后面发送到服务器端，如果有多个字段信息由“&”分开。举例如下：

```
http://127.0.0.1 :8080/plugin.jsp? txtName="…"& txtNum="…"
```

服务器端应用 request 对象的方法来接受或处理这些信息。request 对象用得比较多的是 getParameter 方法。

8.2.2 getParameter 方法

1. getParameter 方法的作用

request 对象的 getParameter 方法根据指定的参数获取客户端信息。

2. getParameter 方法的语法规则

```
<% String name[] %>
…
<% name=request.getParameter("txtName")%>
```

name 是一个声明过的字符串变量。txtName 是客户端提交信息中的一个字段名。

8.2.3 获取客户提交信息案例

例 8.1 要求在页面上有两个文本框，在文本框中输入姓名和电话号码，单击“提交”按钮后，由服务器应用程序接受并处理用户信息。例 8.1 中 ex8-01.html 通过表单向 ex8-01.jsp 提交信息。

ex8-01.html 代码清单如下：

```
<html>
<head><title>获取客户提交信息案例</title>
</head>
<body>
  <div align="center">
   <form action="ex8-01.jsp" method=post>
       <font size=4 color=blue>
       姓名：<input name=RdName><p>
       电话：<input name=PhName><p>
      <input type="submit" value="提交"name="submit"></font>
   </form>
  </div>
</body></html>
```

ex8-01.jsp 代码清单如下：

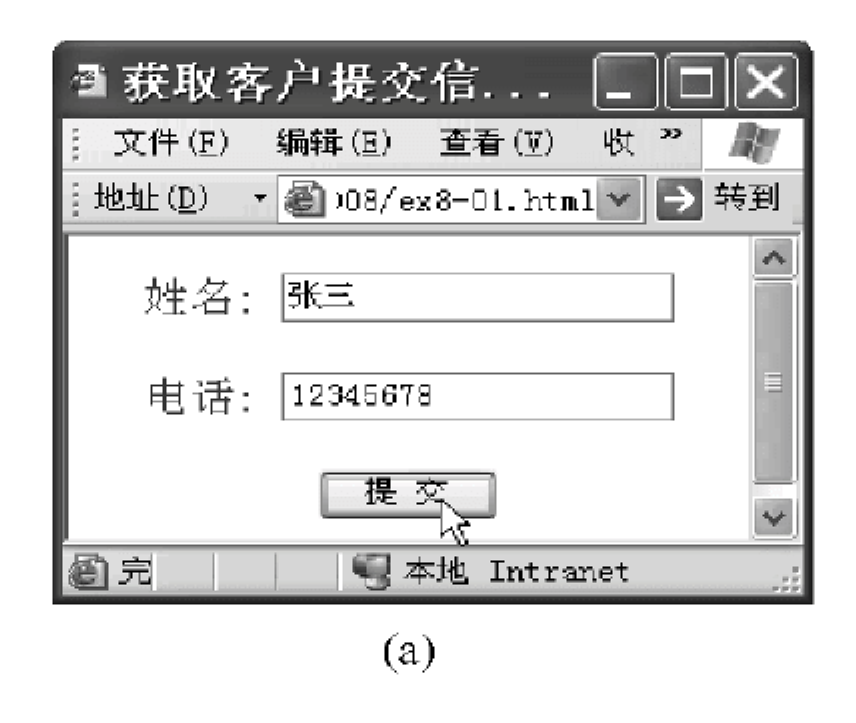
```
<%@ page contentType="text/html;Charset=GB2312"%>
<html><body>
  <% String str1=request.getParameter("RdName");%>
  <% String str2=request.getParameter("PhName");%>
  <% String str3=request.getParameter("submit");%>
  <font face="楷体" size=4 color=blue>
  姓名文本框信息：<%=str1%><p>
  电话文本框信息：<%=str2%><p>
  提交按钮的面值：<%=str3%>
  </font>
</body></html>
```

例 8.1 运行结果如图 8-1 所示。

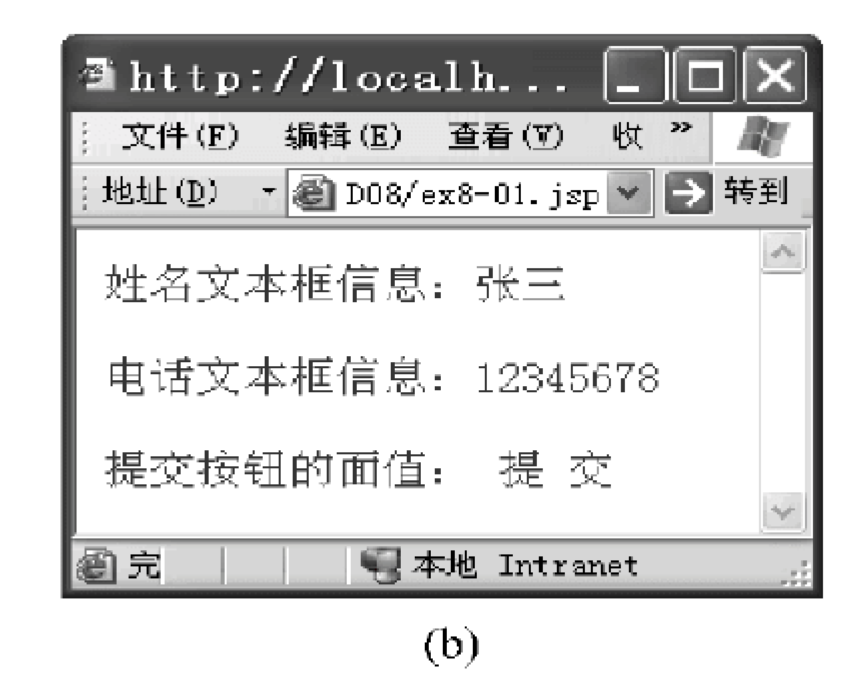
8.2.4 request 对象常用方法

在客户访问页面时，向服务器的 JSP 引擎提交一个 HTTP 请求，该请求包含一个请求行、主机头和上载信息，例如：

```
post/ex8-01.html/HTTP 1.1   //请求行
```



(a)



(b)

图 8-1　获取客户提交信息案例

host:local:8080　//头,host 是头名字

上载的表单信息是用 getParameter 方法获得的(见例 8.1)。request 对象也提供了一些其他方法,用来获得客户传来的信息,具体如下。

(1) getProtocol()方法:获得客户向服务器传送数据所使用的通信协议和它的版本号,例如,HTTP/1.1。

(2) getServerName()方法:获得接受请求的服务器主机名。

(3) getServerPort()方法:获得服务器主机的端口号。

(4) getRemoteHost()方法:获得客户机的全名,如果不能获取客户机名字,则获得客户机的 IP 地址。

(5) getRemoteAddr()方法:获得发送请求的客户机的 IP 地址。

(6) getMethod()方法:获得客户提交信息的方式,如 get、post 或 put 等。

(7) getServletPath()方法:获得客户请求的 JSP 页面的文件目录。

(8) getContentLength()方法:获得客户提交信息的长度,以字节为单位。

(9) getHeader(String name)方法:获得 HTTP 头文件中由参数 name 指定的头名字的值。

(10) getHeaderNames()方法:获得客户请求中所有头部域的名字。

(11) getPathInfo()方法:获得客户请求时关联到 URL 的附加路径信息,没有此信息则返回空值。

(12) getCookies()方法:返回客户端的 Cookies 对象,结果是一个 Cookies 数组。如果浏览器没有发送 Cookies,则返回空值。

(13) getRequestURL()方法:获得发出请求字符串的客户端地址。

(14) getParameter(String name)方法:获得客户提交给服务器的 name 参数值。

(15) getParameterNames()方法:获得客户提交给服务器的所有参数名。

(16) getParameterValues(String name)方法:获得指定参数所有值。

8.2.5　request 对象常用方法应用案例

例 8.2　使用 request 对象的常用方法,获取客户端提交的各种信息。例 8.2 把例 8.1 的文件 ex8-01.html 的 form 的 action 属性改为 ex8-02.jsp 即可,更改如下:

```
<form method="post" action="ex8-02.jsp">
```

ex8-02.jsp 代码获取了客户端的信息，并把它们显示出来，代码清单如下：

```
<%@ page contentType="text/html;Charset=GB2312"%>
<html><head><title>request 对象常用方法应用案例</title></head>
<body>
<%
   out.println("姓名文本框提交信息："+request.getParameter("RdName")+"<br>");
   out.println("电话文本框提交信息："+request.getParameter("PhName")+"<br>");
   out.println("客户端协议名和版本号："+request.getProtocol()+"<br>");
   out.println("客户机名："+request.getRemoteHost()+"<br>");
   out.println("客户机的 IP 地址："+request.getRemoteAddr()+"<br>");
   out.println("客户提交信息的长度："+request.getContentLength()+"<br>");
   out.println("客户提交信息的方式："+request.getMethod()+"<br>");
   out.println("HTTP 头文件中的 Host 值："+request.getHeader("Host")+"<br>");
   out.println("服务器名："+request.getServerName()+"<br>");
   out.println("服务器端口号："+request.getServerPort()+"<br>");
   out.println("客户请求页面的文件目录："+request.getServletPath()+"<br>");
%>
</body></html>
```

运行结果如图 8-2 所示。

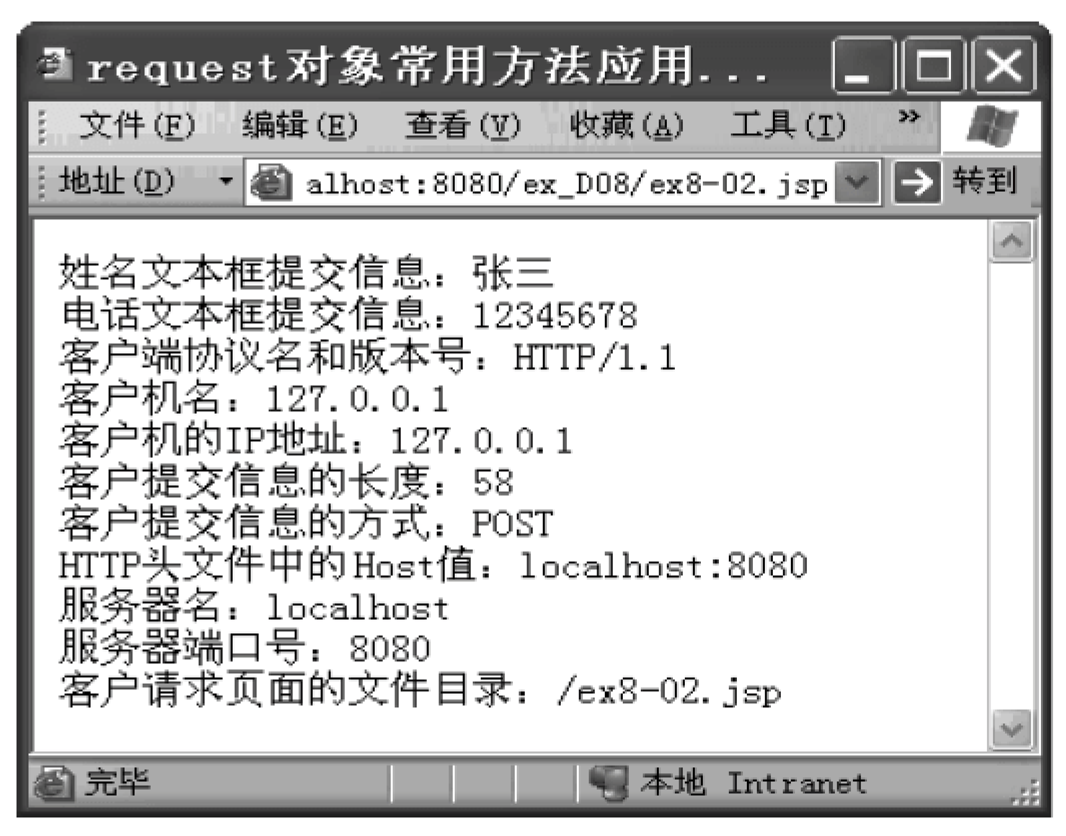


图 8-2　request 方法的输出

8.3　response 对象

8.3.1　response 对象的功能

response 对象把服务器端的数据以 HTTP 的格式发送到客户端浏览器。response 的功能与 request 对象的功能相反，request 对象用于得到用户提交的信息，而 response

对象是向用户发送信息，两者结合起来完成动态页面的交互功能。

response 对象提供了几种输出结果的方法，例如：页面重新定向(sendRedirect)方法、设置状态行(setStatus)方法和设置文本类型(setContentType)方法等。

8.3.2 sendRedirect 方法

1. sendRedirect 功能

sendRedirect 方法将客户端浏览器转向新的页面，页面中重定向语句后的代码不再被执行。它可以根据用户的不同要求转向不同的页面。

2. 应用举例

例 8.3 显示用户登录界面，并验证登录者输入的姓名和密码是否完整，如果不完整转向新的页面请用户重新输入，如果输入完整则显示输入的姓名和密码。例 8.3 由三个程序组成。

- ex8-03.html：登录界面，见图 8-3，登录者输入姓名和密码。
- ex8-03.jsp：接受表单 ex8-03.html 的信息，判断输入是否完整，若输入完整，显示登录成功界面(如图 8-4(a)所示)；如果输入不完整，由 response.sendRedirect("ex8-03_1.html")语句将页面重新定向到异常处理页面 ex8-03_1.html，要求读者重新输入。



图 8-3 登录界面

- ex8-03_1.html：输入数据不完整，提示读者重新输入，异常处理界面见图 8-4(b)。



(a) 登录成功界面



(b) 异常处理界面

图 8-4 登录成功和异常处理界面

① ex8-03.html 代码清单如下：

```
<html>
<head><title>sendRedirect 方法应用案例</title>
```

```
</head>
<body><center>
<font size="5" face="隶书" color="#ff0000">欢迎登录</font><hr>
<form action="ex8-03.jsp" method=post><font size=4>
    姓名：<input name=RdName size=20><br>
    密码：<input type="password" name=RdPasswd size=21><p>
    <input type="submit" value="确定" name="submit">
    <input type="reset" value="清除" name="reset"></font>
</form></center>
</body></html>
```

② ex8-03.jsp 代码清单如下：

```
<%@ page contentType="text/html;Charset=GB2312"%>
<%
  String Name=request.getParameter("RdName");
  String Passwd=request.getParameter("RdPasswd");
  if(Name.equals("") ||Passwd.equals(""))
    response.sendRedirect("ex8-03_1.html");
%>
<html>
<head><title>sendRedirect 方法应用案例</title>
</head>
<body><center>
  <font size="5" face="隶书" color="#ff0000">登录成功</font></center><hr><P>
  姓名是：<%=Name %><P>
  密码是：<%=Passwd%>
</body></html>
```

③ ex8-03_1.html 代码清单如下：

```
<html>
<head><title>sendRedirect 方法应用案例</title>
</head>
<body>
<center><font size="5"face="隶书"color="#ff0000">信息不完整，请重新输入！</font><hr>
<form action="ex8-03.jsp" method=post><font size=4>
  姓名：<input name="RdName" size=20><br>
  密码：<input type="password" name=RdPasswd size=21><p>
  <input type="submit" value="确定 " name="submit">
  <input type="reset" value="清除 " name="reset"></font>
</form></center>
</body></html>
```

8.3.3 response 的状态行

1. 状态行作用

当页面出现错误时，服务器会自动响应，将相应的出错信息返回客户端。状态行包含 3 位数字的状态代码，代表错误的性质和处理方法。共有 5 类状态码。

(1) 1XX(1 开头的 3 位数)：主要是实验性质的。例如 101 表示服务器正在升级协议。

(2) 2XX：表示请求成功。如 200 表示请求成功。

(3) 3XX：表示在请求满足之前应采取的进一步行动。如 305 表示请求必须通过代理来访问。

(4) 4XX：浏览器不能满足请求时返回的状态码。如 404 表示请求的页面不存在。

(5) 5XX：服务器执行出现错误时返回的状态码。如 500 表示服务器内部发生错误，不能服务。

程序中可以使用 response 对象的 setStatus()方法设置状态行。但是在一般情况下，不需要在程序中设置状态码。当页面出现问题时，服务器会自动响应，并发送相应的状态码提示用户。

2. setStatus( )方法

使用 response. setStatus(int n) 方法设置状态行，例如 response. setStatus(501)取得错误信息为 501 的出错信息，返回该出错页面到客户端。如果状态代码为出错码，页面中 response. setStatus()后面的语句将不被执行。

3. 状态行应用案例

例 8.4 本案例包含 4 个文件：ex8-04. html、ex8-04_1. jsp、ex8-04_2. jsp 和 ex8-04_3. jsp，在 ex8-04. html 页面单击不同的超链接，将显示不同的状态码状态。

(1) ex8-04. html 代码清单如下：

```
<html>
<head><title>response 对象状态行应用案例</title>
</head>
<body>
<center>
  <font size="5" face="隶书" color=blue>显示不同的状态行</font><hr>
  <font size="3" face="楷体" color=green>
    <a href="ex8-04_1.jsp">200 请求成功信息</a><p>
    <a href="ex8-04_2.jsp">404 请求资源不可用信息</a><p>
    <a href="ex8-04_3.jsp">501 不支持请求的部分功能</a><p>
```

```
  </font>
</center>
</body></html>
```

(2) ex8-04_1.jsp 代码清单如下：

```
<%@ page contentType="text/html;Charset=GB2312"%>
<html>
<head><title>response 对象状态行应用案例</title>
</head>
<body>
<center>
   <font size="5" face="隶书" color=blue>200 请求成功信息</font><br><hr>
   <% response.setStatus(200);
        out.println("一切正常");
   %>
</center>
</body></html>
```

(3) ex8-04_2.jsp 代码清单如下：

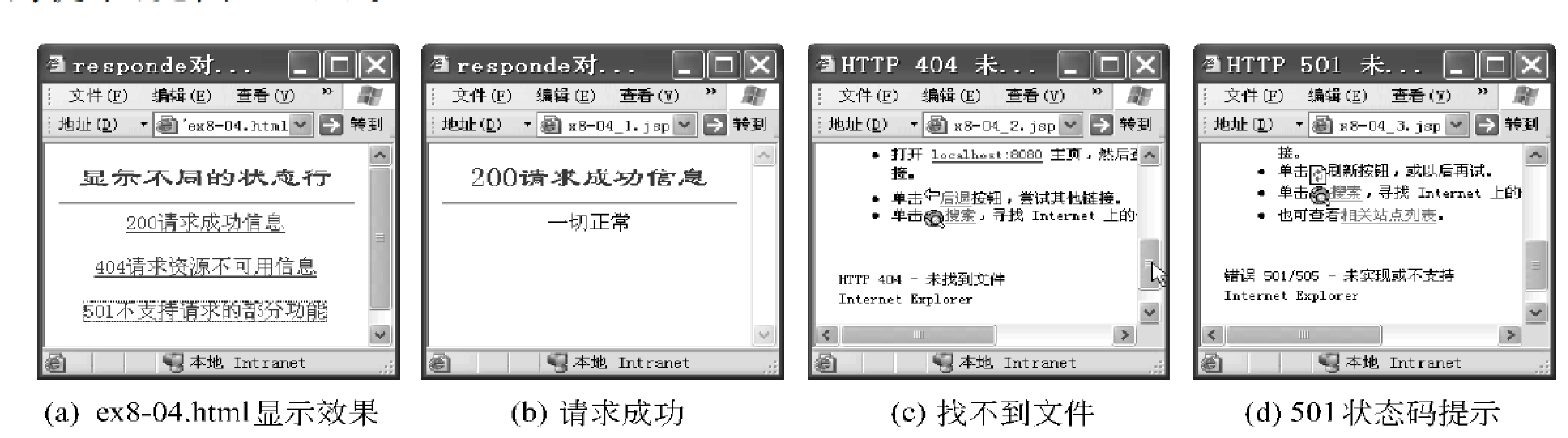
```
<%@ page contentType="text/html;Charset=GB2312"%>
<html>
<head><title>response 对象状态行应用案例</title>
</head>
<body>
<center>
   <font size="5" face="隶书" color=blue>404 请求资源不可用信息</font><br><hr>
   <% response.setStatus(404);
        out.println("不能显示");
   %>
</center>
</body></html>
```

(4) ex8-04_3.jsp 代码清单如下：

```
<%@ page contentType="text/html;Charset=GB2312"%>
<html>
<head><title>response 对象状态行应用案例</title>
</head>
<body>
<center>
   <font size="5" face="隶书" color=blue>501 不支持请求的部分功能</font><br><hr>
   <% response.setStatus(501);
        out.println("不能显示");
   %>
</center>
```

</body></html>

(5) ex8-04.html 的显示效果如图 8-5(a)所示，如果在页面中单击“200 请求成功信息”超链接，页面将跳转到 ex8-04_1.jsp 请求成功页面，见图 8-5(b)。如果在页面中单击“501 不支持请求的部分功能”超链接，页面将跳转到 ex8-04_3.jsp 页面，显示 501 状态码的提示，见图 8-5(d)。



(a) ex8-04.html显示效果　(b) 请求成功　(c) 找不到文件　(d) 501 状态码提示

图 8-5　状态码应用案例

8.3.4　setContentType 方法

1. setContentType 方法功能

使用 response 对象的 setContentType()方法可以在程序运行过程中根据需要动态设置 ContentType 属性值。

2. setContentType 语法格式

语法：

response.setContentType(String s);

使用 setContentType(String s)方法动态设置 MIME 类型，参数 s 可以取以下值。

(1) text/html：HTML 超文本文件，后缀为.html。

(2) text/plain：plain 文本文件，后缀为.txt。

(3) application/msword：Word 文档文件，后缀为.doc。

(4) application/x-msexcel：Excel 表格文件，后缀为.xls。

(5) image/jpeg：jpeg 图像，后缀为.jpeg。

(6) image/gif：gif 图像，后缀为.gif。

3. setContentType 方法应用案例

例 8.5　根据要求选择使用 HTML 类型、Word 类型或 Excel 类型，显示同一个 dataTxt.txt 文件。在 ex8-05.html 页面上选择不同按钮，然后单击“提交”按钮，由ex8-05.jsp 文件选择显示类型。

(1) ex8-05.html 代码清单如下:

```
<html>
<head><title>setContentType 方法应用案例</title></head>
<body>
<form action="ex8-05.jsp" method="get">
  <font size="5" face="隶书" color=blue>请选择文件显示类型</font><hr>
  <input type="radio" name="showType" value="0">HTML 类型显示<br>
  <input type="radio" name="showType" value="1">Word 类型显示<br>
  <input type="radio" name="showType" value="2">Excel 类型显示<br>
  <input type="submit" name="submit" value="提交">
</form>
</body></html>
```

(2) ex8-05.jsp 代码清单如下:

```
<%@ page contentType="text/html;Charset=gb2312"%>
<html>
<head><title>setContentType 方法应用案例</title></head>
<body>
<% String str=request.getParameter("showType");
  if(str==null){str="";}
  else{
    if(str.equals("0")){
      response.setContentType("text/html;charset=gb2312");}
    else if(str.equals("1")){
          response.setContentType("application/msword;charset=gb2312");}
        else{
          response.setContentType("application/x-msexcel;charset=gb2312");}
    }
%>
<jsp:include page="dataTxt.txt" flush="true"/>
</body></html>
```

(3) dataTxt.txt 文件如下:

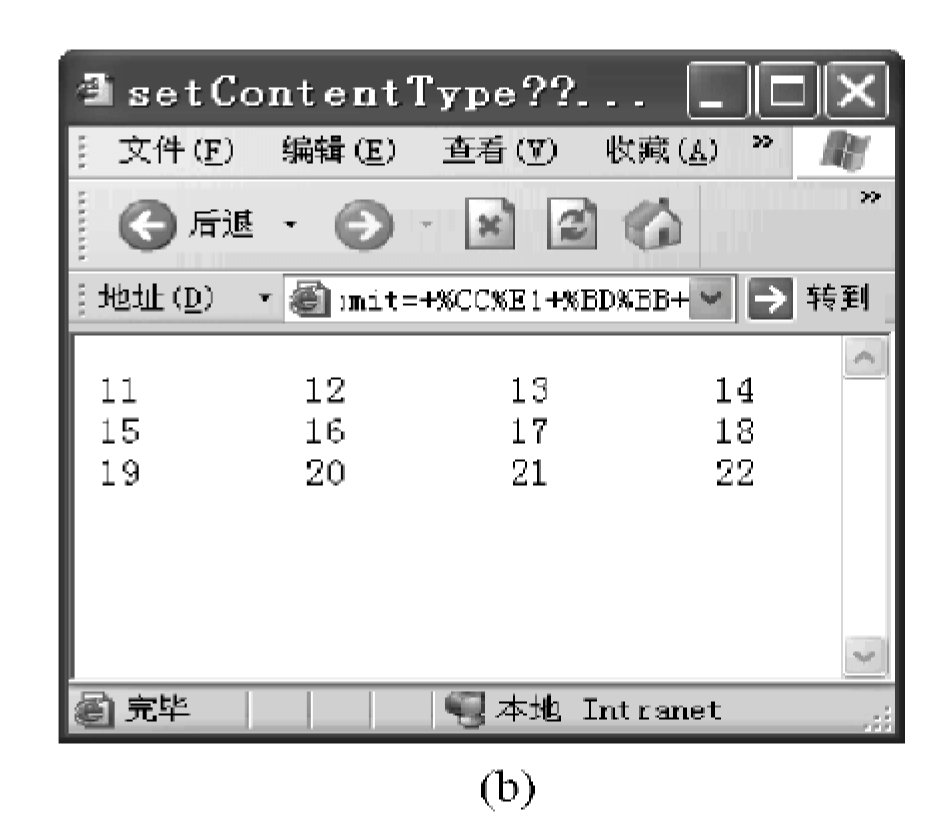
```
11          12          13          14<br>
15          16          17          18<br>
19          20          21          22<br>
```

注意:因为浏览器视半角输入的多个空格为一个空格,所以如果希望用 Excel 显示该文件,输入 dataTxt.txt 文件中的空格时,需要把输入法切换到全角。

(4) ex8-05.html 在浏览器中的显示效果如图 8-6(a)所示,如果在页面中选择 HTML 类型显示,如图 8-6(b)所示;选择 Word 类型显示,如图 8-7(a)所示;选择 Excel 类型显示,如图 8-7(b)所示。

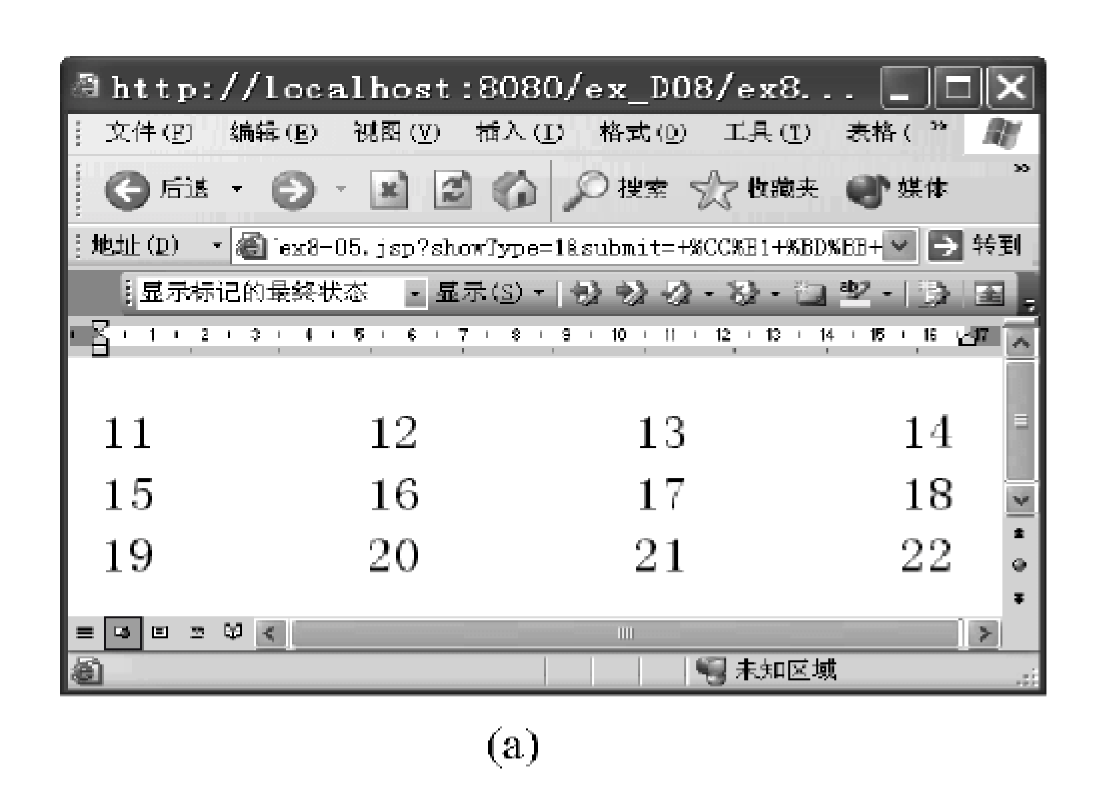


(a)

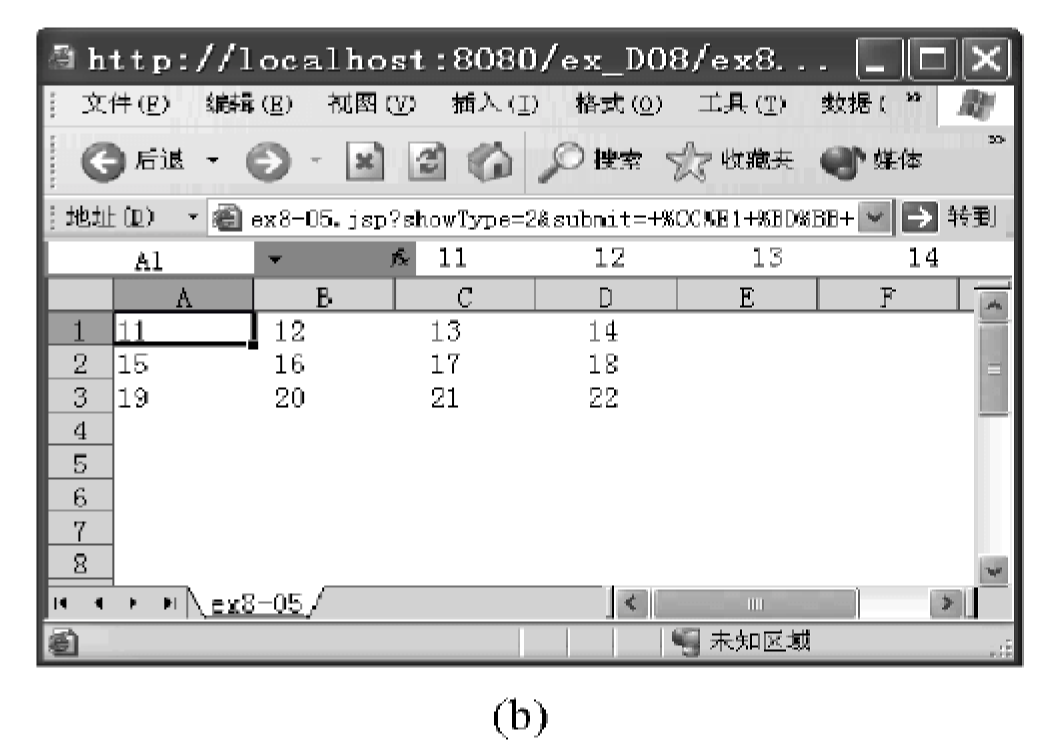


(b)

图 8-6　HTML 类型



(a)



(b)

图 8-7　Word 类型和 Excel 类型显示

8.3.5　response 对象的其他方法

response 对象的其他常用方法如下。

(1) getCharacterEncoding()方法：返回服务器响应客户所使用的编码属性。

(2) getWriter()方法：获得一个打印输出对象，向客户发送文本。

(3) setContentLength()方法：设置服务器发送给客户端内容的长度。

(4) setHeader()方法：动态添加新的响应头和头的值。

8.3.6　response 方法应用案例

例 8.6　代码 ex8-06.jsp 返回服务器的编码属性和一个打印输出对象，并每 8 秒钟刷新一次页面，显示系统时间。代码清单如下：

```
<%@ page contentType="text/html; Charset=GB2312"%>
<%@ page import="java.util.*"%>
<html>
```

```
<head><title>response 对象常用方法应用案例</title></head>
<body><font face="楷体" size=3 color=blue>
<%
   out.println("服务器使用的编码属性："+response.getCharacterEncoding()+"<br>");
   out.println("打印输出对象："+response.getWriter()+"<br>");
%><hr>
   页面每 8 秒钟刷新一次
   现在的时间是：<br>
<%
   out.println(""+new Date());
   response.setHeader("Refresh","8");
%></font>
</body></html>
```

运行结果如图 8-8 所示。



图 8-8　刷新页面

8.4　out 对象

8.4.1　out 对象的功能

out 对象是一个输出流，用来向客户端发送数据。out 对象发送的内容具有文本的性质。可以通过 out 对象直接向客户端发送一个由程序动态生成的 HTML 文件。常用的方法有 print 和 println。由于 out 对象内部包含了一个缓冲区，所以需要一些对缓冲区进行操作的方法，例如 clear、clearBuffer、flush、getBufferSize 和 getRemaining 等。

8.4.2　out 对象中预定义的常量和变量

在某些 out 对象中需要使用常量和变量，变量在使用时需要赋值。out 对象常用属性如下。

（1）NO_BUFFER：表示非缓冲区输出。

(2) DEFAULT_BUFFER：缓冲区输出，使用默认的缓冲区大小。

(3) bufferSize：缓冲区大小，以字节为单位的整型数。

(4) autoFlush：是否自动清空缓冲区。

8.4.3 out 对象方法

(1) print()方法：输出数据。

(2) println()方法：输出数据，并换行。

(3) clear()方法：清除缓冲区中的内容。如果缓冲区被清除过(flush)，则抛出一个 IO 异常，表示数据已经写到客户响应流中。

(4) clearBuffer()方法：清除缓冲区当前的内容。如果缓冲区被清除过(flush)，就不抛出 IO 异常。

(5) flush()方法：把缓冲区的内容写入输出流，并清空缓冲区。

(6) close()方法：关闭流，并在关闭之前进行 flush 操作。一旦流被关闭，就不能再使用 flush()方法。

(7) getBufferSize()方法：返回以字节为单位的缓冲区大小，无缓冲区时返回 0。

(8) getRemaining()方法：返回以字节为单位的未使用的缓冲区大小。

8.4.4 out 对象应用案例

1. out 对象常用方法应用案例

例 8.7 本例说明 out 对象 print/println 方法的应用，ex8-07.jsp 代码清单如下：

```
<%@ page contentType="text/html; Charset=GB2312"%>
<%@ page language="java" autoFlush="false"%>
<html>
<head><title>out 对象 print/println 方法应用案例</title></head>
<body><center><font size=5 color=blue>print/println 方法应用</font>
</center><hr><br>
<%
    out.println("你好!");
    out.clear();
    out.println("after clear: <br>");
    char a='h';
    int m=8;
    double f=3.1415926;
    out.print("a="+a+"");
    out.print("m="+m+"");
    out.print("f="+f+"<br>");
    out.print("BufferSize: "+out.getBufferSize()+"<br>");
```

```
    out.print("Remaining:"+out.getRemaining()+"<br>");
%>
</body></html>
```

代码 ex8-07.jsp 运行结果见图 8-9。由于使用了 clear()方法,在语句"out.clear();"之前的输出从缓冲区中清除掉了,所以没有在浏览器中显示。

2. flush()方法应用

例 8.8 代码 ex8-08.jsp 应用 for 循环延迟文字的输出,并使用 flush()方法把缓冲区的内容输出到页面,使输出的文字逐行显示出来。ex8-08.jsp 代码清单如下:

```
<%@ page contentType="text/html; Charset=GB2312"%>
<html>
<head><title>out 对象 flush()方法的应用</title></head>
<body><center><font size=4 color=blue>逐行显示文字</font></center><hr>
<%
    String strShow="Web 技术应用基础!";              //设定输出的文字
%>
<center><font size=3  face="楷体" color=red>
<%
for(int i=0; i <=6; i++) //通过 for 循环,输出 6 行文字于页面中
{
    for(int j=0; j<200000000; j++)               //利用 for 循环延迟文字的输出
    {  }
    out.println(strShow + "<BR>");               //将字串输出至缓冲区
    out.flush(); //将缓冲区的文字输出至网页
}
%>
</font></center>
</body></html>
```

代码 ex8-08.jsp 的运行结果见图 8-10,文字正在被逐行显示。



图 8-9 out 对象 print/println 方法应用

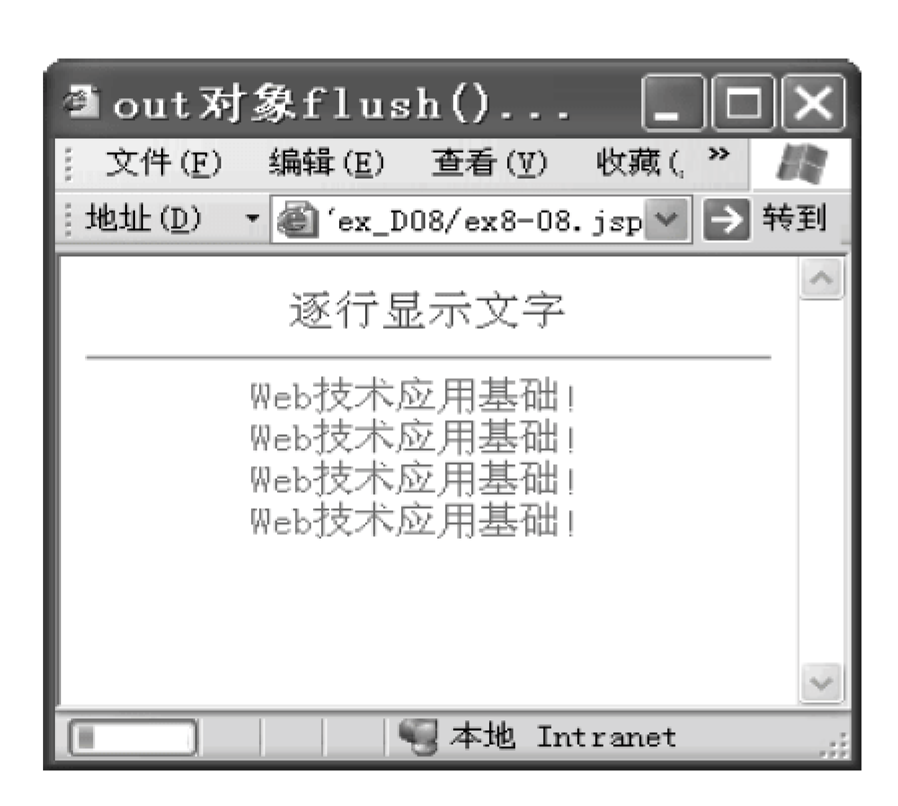


图 8-10 out 对象 flush()方法的应用

8.5 session 对象

8.5.1 会话和会话 ID

从一个客户打开浏览器并连接到服务器开始，一直到这个客户关闭浏览器离开该服务器为止，称为一个会话(session)。HTTP 协议是一种无状态协议，当一个客户向服务器发出请求，请求完成后，连接就被关闭了。服务器不再保留有关连接的信息，所以当下一次请求并连接时，服务器无法判断本次连接和以前的连接是否是同一个客户。但是在很多应用中，需要服务器在用户访问的一个会话期中记住客户，为客户提供连续服务。例如，网上书店，应当允许用户在不同位置挑选不同的书籍，放入购书车，最后再一次性付款。在购书的过程中，服务器应当记住用户的个人信息和已选择的图书，这需要一个在会话过程中一直有效的变量，即会话级变量，记录用户在这段时间内的逻辑上相关联的不同请求。

session 对象提供一个在多个请求之间持续有效的对象，这个对象允许用户存储和提取会话状态信息，它可用于具有多个页面的事务处理。

当一个客户第一次打开浏览器访问服务器上的某个 JSP 页面时，JSP 引擎为客户创建一个 session 对象，用来存储客户在访问各个页面期间提交的各种信息，例如姓名、号码等。同时为该对象分配一个 String 类型的 ID 号，JSP 引擎在响应客户请求的同时，把 ID 号发到客户端，并写入客户端的 cookie 中。如此，session 对象和客户之间建立起一种对应的关系，不同的客户有不同的 session 对象，服务器可以通过不同的 ID 号识别不同的客户。当客户关闭浏览器后，一个会话结束，服务器端该客户的 session 对象被取消。当客户重新打开浏览器建立新连接时，JSP 引擎为该客户再创建一个新的 session 对象。session 对象具有一些方法，用来在会话期间传递信息。

8.5.2 session 对象常用方法

session 对象有如下常用方法。

(1) setAttribute(String name,Object obj)方法：把对象 obj 存入由 name 指定名字的 session 对象。

(2) getAttribute(String 变量名)方法：从 session 对象中取得变量，返回值类型为 Object。若返回值是 null，表示 session 对象中不存在该对象。

(3) getAttributeNames()方法：产生一个枚举对象，该对象调用 nextElements()方法遍历 session 对象中的所有对象。

(4) removeAttribute(String name)方法：删除在 session 对象中由 name 指定的对象。

(5) getCreationTime ()方法：获得 session 对象创建的时间。

(6) getLastAccessedTime ()方法：返回 session 对象最后一次被用户访问的时间，单位是毫秒。

(7) getId ()方法：获得 session 对象的编号 ID。

(8) getMaxInactiveInterval()方法：获得 session 对象的生存时间，单位是秒。

(9) setMaxInactiveInterval()方法：设置 session 对象的生存时间，单位是秒。

(10) getValue (String name，Object value)方法：从 session 对象获得名为 name 的属性值。

(11) putValue (String name，Object value)方法：存储由 name 指定名字的属性值到 session 对象中。

(12) getValueNames ()方法：返回存储在 session 对象中的所有对象名。

(13) invalidate ()方法：删除会话，并清除存储在该对象中的所有对象。

(14) isNew ()方法：如果服务器创建了此会话对象，但客户还未加入，此方法返回值为真。

(15) removeValue (String name)方法：删除 session 中特定名称的对象。

8.5.3 session 对象应用案例

1. 一次会话

例 8.9 本例由三个页面组成：ex8-09.jsp、ex8-09_1.jsp 和 ex8-09_2.jsp，客户在三个页面之间转跳，只要不关闭浏览器，三个页面共享同一个 session 对象，它们的 ID 是一样的。

(1) ex8-09.jsp 代码清单如下：

```
<%@ page contentType="text/html; Charset=GB2312"%>
<html><head><title>会话(session)和会话 ID</title></head>
<body><center>
<font size=3 color=blue>第一个页面的 sessionID 是：<br>
<%
    out.println(session.getId());
%><hr>
<a href=ex8-09_1.jsp>第二个页面</a>
<a href=ex8-09_2.jsp>第三个页面</a>
</font></center>
</body></html>
```

(2) ex8-09_1.jsp 代码清单如下：

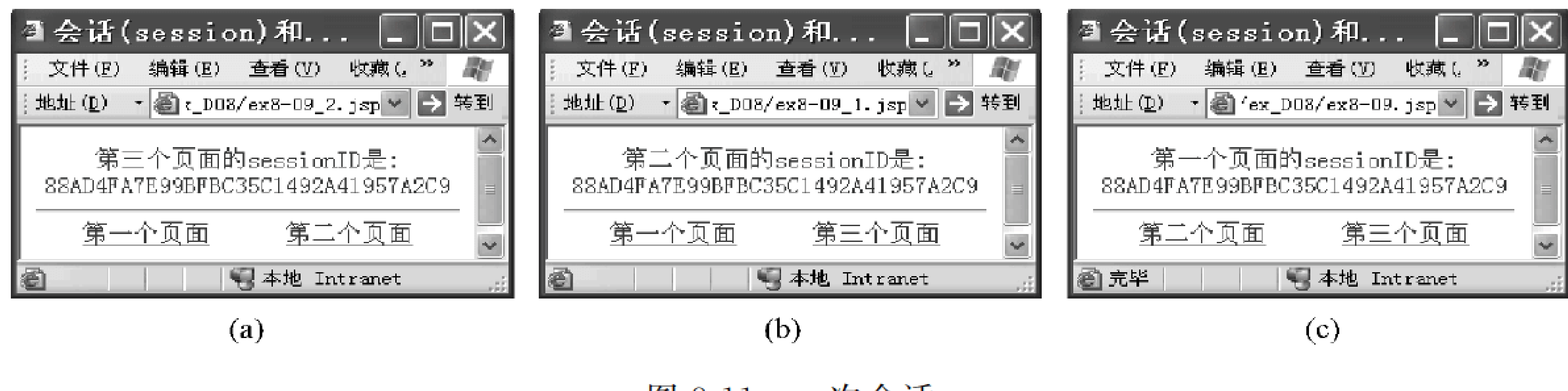
```
<%@ page contentType="text/html; Charset=GB2312"%>
<html><head><title>会话(session)和会话 ID</title></head>
<body><center><font size=3 color=blue>第二个页面的 sessionID 是：<br>
<%
```

```
    out.println(session.getId());
%><hr>
<a href=ex8-09.jsp>第一个页面</a>
<a href=ex8-09_2.jsp>第三个页面</a>
</font></center>
</body></html>
```

(3) ex8-09_2.jsp 代码清单如下：

```
<%@ page contentType="text/html; Charset=GB2312"%>
<html>
<head><title>会话(session)和会话 ID</title>
</head>
<body><center><font size=3 color=blue>第三个页面的 sessionID 是：<br>
<%
    out.println(session.getId());
%><hr>
<a href=ex8-09.jsp>第一个页面</a>
<a href=ex8-09_1.jsp>第二个页面</a>
</font></center>
</body></html>
```

(4) 在浏览器中运行例 8.9 的显示效果如图 8-11 所示。



(a) (b) (c)

图 8-11 一次会话

2. 在会话期传递参数

Web 服务器为每个访问站点的用户创建一个 session 对象，在 session 对象中可以保存多个变量。通过 session.putValue()方法生成 session 中的变量，例如 session.putValue("BookName","Web 技术应用基础!")语句在 session 中保存 BookName 变量，其值为“Web 技术应用基础!”。只要 session 存在，在同一个用户的其他页面中，该 session 中的值仍然可以使用。使用 session.getValue("BookName")语句，取出 BookName 变量中的值。session 的保留时间由系统设置决定。

例 8.10 本例由三个页面组成一个多页面的 Web 应用，它们的源代码是：ex8-10.jsp、ex8-10_1.jsp 和 ex8-10_2.jsp。由于 session 对象在会话期间是一直有效的，所以在 ex8-10.jsp 中由语句 session.putValue("BookName","Web 技术应用基础!")；建立的 session

对象的值“Web 技术应用基础!”,对后续页面 ex8-10_1.jsp 和 ex8-10_2.jsp 也有效。

ex8-10.jsp:获取 session 对象建立时间和最后一次使用时间。并使用 session 对象的 putValue 方法设置一个 session 对象 BookName 的值,该值是“Web 技术应用基础!”。

ex8-10_1.jsp:获取上一页代码即 ex10-10.jsp 中的 session 对象的 BookName 值,并把它输出到页面。同时设置另一个 session 对象的 BookInfo 值。

ex8-10_2.jsp:显示上两个页面的 session 对象 BookName 和 BookInfo 的值,可以看到在会话期间 session 对象是一直有效的。

(1) ex8-10.jsp 代码清单如下:

```
<%@ page contentType="text/html; Charset=GB2312"%>
<%@ page import="java.util.Date"%>
<html><head><title>会话期传递参数</title></head>
<body><center>
<font size=3 color=blue>会话期传递参数</font>
</center><hr>
<%
   Date startTime=new Date(session.getCreationTime());
   Date lastTime=new Date(session.getLastAccessedTime());
   session.putValue("BookName","Web 技术应用基础!");
%>
session 的 ID 是:
<%
     out.println(session.getId());
%><br>
session 建立时间:<br>
<font color=red><%=startTime %></font><br>
最后使用时间:<br>
<font color=red><%=lastTime %></font><br>
<a href='ex8-10_1.jsp'>书名</a>
</body></html>
```

(2) ex8-10_1.jsp 代码清单如下:

```
<%@ page contentType="text/html; Charset=GB2312"%>
<html><head><title> session 对象值传递 </title></head>
<body><center>
<font size=3 color=blue>session 对象值传递</font>
</center><hr><br>
session 的 ID 是:<br>
<%
     out.println(session.getId());
%><br>
     书 名:<%=session.getValue("BookName")%><p>
<%session.putValue("BookInfo","Web 站点构建技术、Web 编程技术和数据库发布技术。");%>
```

```
<a href='ex8-10_2.jsp'> 内容简介 </a>
</center>
</body></html>
```

(3) ex8-10_2.jsp 代码清单如下:

```
<%@ page contentType="text/html; Charset=GB2312"%>
<html><head><title> session 对象值传递 </title></head>
<body><center>
<font size=3 color=blue>session 对象值传递</font>
</center><hr><br>
session 的 ID 是:
<%
    out.println(session.getId());
%><br>
    书名: <%=session.getValue("BookName")%><p>
    内容简介: <%=session.getValue("BookInfo")%>
</center>
</body></html>
```

(4) 运行 ex8-10.jsp 的结果如图 8-12(a)所示。单击图 8-12(a)中"书名"超链接,页面跳转至 ex8-10_1.jsp 页面,如图 8-12(b)所示。图 8-12(a)页面的 session 值"Web 技术应用基础!"在图 8-12(b)页面上仍然有效。单击图 8-12(b)中"内容简介"超链接,页面跳转至 ex8-10_2.jsp 页面,如图 8-12(c)所示。



(a) (b) (c)

图 8-12 session 会话期间不同页面间的值传递

8.6 application 对象

8.6.1 application 对象的功能

application 对象由多个客户端用户共享,它的应用范围是所有的客户。服务器启动后,新建一个 application 对象,该对象一旦建立,就一直保持到服务器关闭。当有客户访

问服务器上的一个 JSP 页面时，JSP 引擎为该客户分配已建立的 application 对象；当客户在所访问站点的不同页面浏览时，他的 application 是同一个，直到服务器关闭。与 session 对象不同的是：不同的客户拥有不同的 session 对象，而所有的客户拥有同一个 application 对象。所以可以用 application 对象保存服务器运行时的全局数据，例如页面的访问次数等。

8.6.2 application 对象常用方法

(1) getAttribute(String name)：返回由 name 指定名字的 application 对象属性的值。

(2) setAttribute(String name,Object obj)：由 obj 来初始化 name 的属性值。

(3) getAttributeNames()：获得一个枚举对象，该枚举对象调用 nextElements()方法可以获得 application 对象中的所有变量名。

(4) getInitParameter(String name)：返回 application 对象中 name 属性的初始值。

(5) getServerInfo()：获得 JSP 引擎名和版本号。

(6) getRealPath()：获得文件的实际路径。

(7) removeAttribute(String name)：删除 application 中的 name 对象。

(8) getMimeType ()方法：返回特定文件的 MIME 类型，如果是未知的 MIME 类型则返回空值。常见的 MIME 类型有："txt/html"和"image/gif"。

8.6.3 application 对象应用案例

1. 输出全局数据

例 8.11 本例的源代码 ex8-11.jsp 使用 application 对象输出应用程序在服务器运行期间的一些全局信息，代码清单如下：

```
<%@ page contentType="text/html; Charset=GB2312"%>
<html><head><title>application</title></head>
<body>
<%
  out.println("Java Servlet API 版本号："+application.getMajorVersion()+"."+
             application.getMinorVersion()+"<br>");
  out.println("文件 login.htm 的 MIME 类型："+application.getMimeType("\\login.htm")+"<br>");
  out.println("程序 ex8-11.jsp 的 URL："+application.getRealPath("ex8-11.jsp")+"<br>");
  out.println("服务器信息："+application.getServerInfo()+"<br>");
  application.setAttribute("BookName","Web 技术应用基础");
  out.println("书名 ："+application.getAttribute("BookName")+"<br>");
%>
</body></html>
```

代码 ex8-11.jsp 的运行结果如图 8-13 所示。



图 8-13 application 对象应用

2. 计数器

例 8.12 ex8-12.jsp 代码对用户的访问量进行统计，代码清单如下：

```
<%@ page contentType="text/html; Charset=GB2312"%>
<%
if(session.isNew()){        //同步处理
  synchronized(application){
    Integer count=(Integer)application.getAttribute("count");
    if (count==null){
       count=new Integer(1);
       application.setAttribute("count",count);
    }
    else {
       count=new Integer(count.intValue()+1);
             //将访问量保存到内存当中
       application.setAttribute("count",count);
    }
    out.print("您是本站的第"+count+"位客人");
  }
}
%>
```

代码 ex8-12.jsp 的运行结果如图 8-14 所示。





图 8-14 计数器

8.7 exception 对象

8.7.1 exception 对象的功能

exception 对象用来发现、捕获和处理异常，它是 JSP 文件运行异常时产生的对象。当 JSP 文件运行时如果有异常发生，则抛出异常，该异常只能被使用了＜%@ page isErrorPage="true"%＞的JSP 文件捕获。

JSP 异常处理机制见图 8-15。如果 JSP 文件在运行时有异常现象发生，则抛出一个异常。如果该页中定义了异常处理页，则由异常处理页来处理异常，如果没有定义异常处理页则由服务器处理异常。

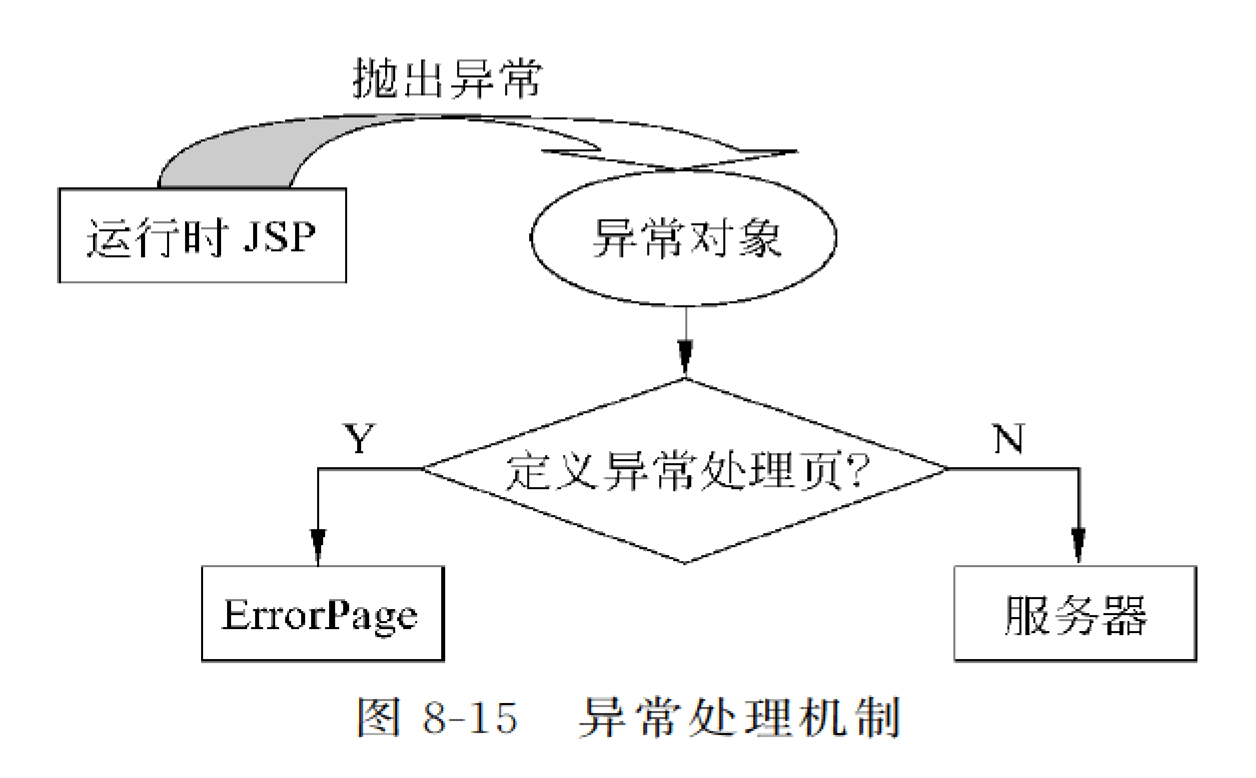


图 8-15　异常处理机制

8.7.2 JSP 异常处理语句

应用 try-catch-finally 语句进行异常处理，结构如下：

```
try{
  抛出异常模块
}
catch{
  捕获异常模块
}
finally{
  处理异常模块
}
```

8.7.3 exception 对象常用方法

（1）getMessage 方法：获取异常信息。

（2）toString 方法：获取该异常对象的简短描述。如果该对象包含异常消息字符串，则返回数据“对象的实际类名＋“：”＋ getMessage 方法的返回值”，如果该对象不包含异常消息字符串，则返回实际的类名。

8.7.4 异常处理应用案例

例 8.13　本例包含两个程序 ex8-13.jsp 和 ex8-13_1.jsp。

ex8-13.jsp：代码 ex8-13.jsp 中的除法用 0 做了除数，抛出一个异常，该异常由 ex8-13_1.jsp 异常处理页去处理。

ex8-13_1.jsp：处理异常。

(1) ex8-13.jsp 代码清单如下：

```
<%@ page  errorPage="ex8-13_1.jsp" contentType="text/html;Charset=GBK"%>
<%
  int a=10,b=0, c;
  try{
    c=a/b;
    out.print(c);
  }
  catch(RuntimeException rtex){
    throw new RuntimeException("除数不能为 0!");
  }
%>
```

(2) ex8-13_1.jsp 代码清单如下：

```
<%@ page isErrorPage="true" contentType="text/html;charset=GBK"%>
<%@ page import="java.util.Exception"%>
<font size=4 color=blue>处理异常页面</font><hr>
<%=exception.toString()%>
```

(3) 代码 ex8-13.jsp 的运行结果如图 8-16 所示。



图 8-16　异常处理

8.8　JSP 其他内置对象

8.8.1　page 对象

1. page 对象的功能

得到正在运行的由 JSP 文件产生的类对象。page 对象应用不多，所以此处只做简单介绍。

2．page 对象的方法

（1）getClass()：获得对象运行时的类。
（2）hashCode()：获得该对象的哈希码值。
（3）equals()：用来判别其他对象是否与该对象相等。
（4）clone()：创建并返回当前对象的复制。
（5）toString()：取得表示该对象的字符串。

3．应用举例

例 8.14 本例 ex8-14.jsp 的源代码说明了 page 对象部分方法的应用，代码清单如下：

```
<%@ page contentType="text/html; Charset=GB2312"%>
<html><head><title>page 对象应用 </title></head>
<body><center>
<font size=4 color=blue>page 对象应用案例</center><hr></font>
<%
    out.println("JSP 文件的类是："+"<br>"+page.getClass()+"<p>");
    out.println("page 对象的哈希码值是："+page.hashCode()+"<p>");
    out.println("page 对象转换成字符串："+page.toString());
%>
</body></html>
```

代码 ex8-14.jsp 的运行结果如图 8-17 所示。

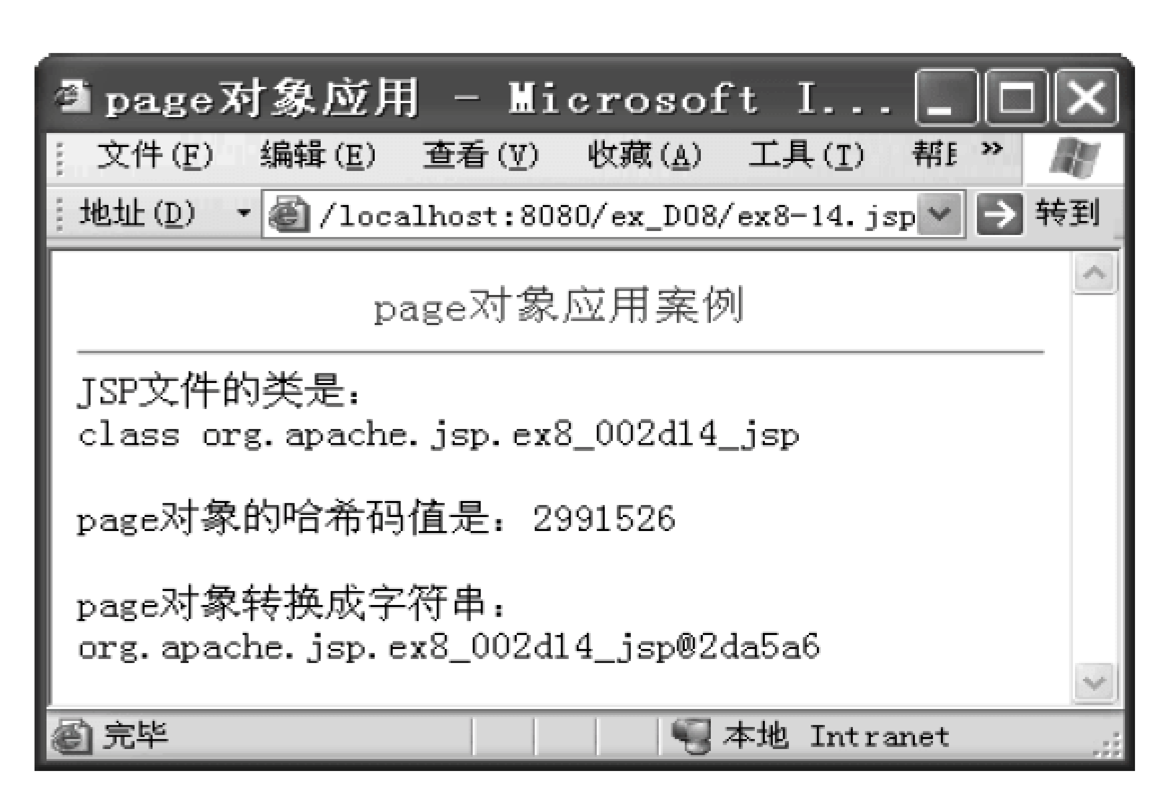


图 8-17 page 对象应用案例

8.8.2 pageContext 对象

1．pageContext 对象的功能

一般常用 pageContext 对象获取当前页面运行的一些属性。

2. pageContext 对象方法

(1) setAttribute(String name,Object obj):由 obj 初始化 name 属性。

(2) getAttribute(String name):返回由 name 指定名字的 pageContext 对象属性值。

(3) findAttribute(String name):按照 page、request、session 和 application 的次序获得给定名字的属性。

(4) removeAttribute(String name):删除 name 属性。

(5) getAttributesScope (String name):返回一个整型数,表示与给定名字相关对象的作用范围。

(6) include(String name):将给定名字包含进来。

3. pageContext 对象应用案例

例 8.15 本例 ex8-15.jsp 说明了 pageContext 对象部分方法的使用。语句:pageContext.setAttribute("文件名",filename);把“文件名”属性指定为 filename(即 ex8-15.jsp)值,保存在 pageContext 对象中。通过 getAttribute 和 findAttribute 方法都可以获得该属性。应用 removeAttribute 方法删除“文件名”属性,删除后再应用 getAttribute 方法,将得不到属性值。使用 include 方法把代码“ex8-14.jsp”包含进来,并在浏览器中输出。ex8-15.jsp 代码清单如下:

```
<%@ page contentType="text/html; Charset=GB2312"%>
<html><head><title> pageContext 对象应用案例 </title></head>
<body><center>
<font size=3 color=blue> pageContext 对象应用案例 </font></center><hr>
<%
  String filename="ex8-15.jsp";
  pageContext.setAttribute("文件名",filename);
  out.println("pageContext.getAttribute(\"文件名\")="+pageContext.getAttribute("文件名")
                                                    +"<br>");
  out.println("pageContext.findAttribute(\"文件名\")="+pageContext.findAttribute("文件名")
                                                    +"<br>");
  out.println("pageContext.getAttributesScope(\"文件名\")=
                                "+pageContext.getAttributesScope("文件名")+"<br>");
  pageContext.removeAttribute("文件名");
  out.println("After remove,the pageContext.getAttribute(\"文件名\")=
                                     "+pageContext.getAttribute("文件名")+"<p>");
  pageContext.include("ex8-14.jsp");
%>
</body></html>
```

代码 ex8-15.jsp 的运行结果如图 8-18 所示。



图 8-18 pageContext 对象应用案例

8.8.3 config 对象

1. config 对象功能

使用 config 对象获取初始化配置信息。

2. config 对象方法

(1) getServletContext():获得 Servlet 与服务器交互的信息。

(2) getInitParameter(String name):获得初始化参数值,如果参数不存在,返回空值。

(3) getInitParameterNames (String name):获得初始化参数名称,如果参数不存在,则返回空值。

config 对象方法的应用与其他对象类似,此处不再赘述。

8.9 Cookie

8.9.1 Cookie 功能

Cookie 的英文原意是"甜点"的意思,当客户访问服务器时,服务器在客户的硬盘上建立一个小文本文件,用来跟踪访问 Web 站点的用户。它记录了有关用户的信息,如:身份证号码、密码、用户购物方式和访问站点次数等。当客户下次再访问同一 Web 站点时,站点的页面会查找这个 Cookie,浏览器把它原样传送给服务器。每个 Web 站点都有自己的 Cookie,它的内容由 Web 服务器管理者决定,可以随时读取,但只能被该 Web 站点的页面读取。Cookie 文件存放在 Windows 的 Cookie 文件夹下。

8.9.2 Cookie 属性

Cookie 对象的属性及其作用见表 8-2。

表 8-2 Cookie 对象的属性

属 性	说 明
name	Cookie 对象的名字，是每个 Cookie 对象必须有的属性
value	Cookie 的值，也是每个 Cookie 对象必须有的属性
expires	Cookie 的过期时间
domain	设置 Cookie 的 Web 页面所在的计算机域名
path	可以设定一个 Cookie 只针对站点的某一层次。该项是可选项，若指定，则 Cookie 只被发送到 path 指定路径的请求中去
secure	是一个布尔值，默认值是 false。如果设为 true，浏览器认为该 Cookie 是安全的服务器

8.9.3 创建 Cookie 对象

创建 Cookie 对象的语法规则如下：

Cookie 对象名＝new Cookie(“变量名”，数值)

例如，Cookie cookieBookNm＝new Cookie("BookName"，"Web 技术应用基础！")；创建了名为 cookieBookNm 的 Cookie 对象，并把数值“Web 技术应用基础！”赋给变量“BookName”。

8.9.4 Cookie 方法

1. addCookie 方法

addCookie 方法向客户机添加一个 Cookie 对象。

2. getCookie 方法

getCookie 方法取得 Cookie 对象的数据。

8.9.5 Cookie 应用案例

例 8.16 Cookie 数据存取。代码 ex8-16.jsp 把字符串“欢迎学习 Web 技术应用基础！”赋给变量“MyString”，并保存到 Cookie 对象 cookieMyString 中，然后再从

cookieMyString 对象中把数据取出。ex8-16.jsp 代码清单如下：

```
<%@ page import="java.util.Date"%>
<%@ page contentType="text/html; charset=GB2312"%>
<html><head><title>Cookie 数据的存取</title></head>
<body><center>
<font size=4 color=blue>Cookie 数据存取案例</font></center><hr>
<% //创建 Cookie 对象
    Cookie cookieMyString=new Cookie("MyString",
      java.net.URLEncoder.encode("欢迎学习 Web 技术应用基础!"));
    Cookie temp=null;
    response.addCookie(cookieMyString);
      //将 Cookie 变量的数值加入 Cookie 对象中
    Cookie[] cookies=request.getCookies();
    int cookielen=cookies.length; //取得 Cookie 数组的长度
    if(cookielen !=0){
      for(int i=0; i<cookielen; i++){//取得 Cookies 数组中的 Cookie 变量
        temp=cookies[i];
        if(temp.getName().equals("MyString")){
%>
          Cookie 中<font color=blue>MyString</font>变量值为 :
          <center><font size=4 color=red>
          <%=java.net.URLDecoder.decode(cookieMyString.getValue())%>
          </font><BR></center>
<%
        }
      }
    }else{
%>      无法取得 Cookie<BR>
<%
    }
%>
</body></html>
```

代码 ex8-16.jsp 的运行结果如图 8-19 所示。



图 8-19　Cookie 数据存取

8.10　JSP 内置对象在网上书店中的应用案例

例 8.17　使用 JSP 内置对象制作网上书店中的订单处理模块。为了使代码易于理解，暂时去除了源代码中与数据库的连接部分。与数据库连接部分的内容将在下一章讲解。

1．任务要求

要求在页面上为客户制作一个订单，接收客户的姓名、地址、联系电话和邮编等信息，订单处理界面如图 8-20 所示。



图 8-20　订单处理

客户在文本框中输入姓名、地址、联系电话和邮编，然后单击"我要下订单"按钮，将表单输入信息提交服务器应用程序来处理。应用程序接收信息后，为用户反馈必要的信息。

2．制作表单，收集用户信息

页面 order1.jsp 代码清单如下：

```
<%@ page contentType="text/html; charset=GB2312"%>
<% request.setCharacterEncoding("GB2312");%>
<script language="javascript">
<! --
function CheckSubmit()
 {
  if( document.orderform.receiver.value=="")
     { alert("请输入收货人姓名!"); document.orderform.receiver.focus(); return false; }
```

```
  if( document.orderform.address.value=="")
     { alert("请输入收货人地址!"); document.orderform.address.focus(); return false; }
  if( document.orderform.phone.value=="")
     { alert("请输入联系电话!"); document.orderform.phone.focus();return false; }
  if( document.orderform.postcode.value=="")
     { alert("请输入邮编!"); document.orderform.postcode.focus();return false; }
  return true;
 }
</script>
<link href="maincss.css" rel="stylesheet" type="text/css">
<div align="center">
<table width="750" border="0" cellspacing="1" cellpadding="1">
  <tr>
    <td ><div align="center"><%@include file="top.jsp"%></div></td>
  </tr>
  <tr>
    <td ><div align="center">
      <table width="100%" border="0" cellpadding="0" cellspacing="0" class="td">
       <form name="orderform" action="order2.jsp" method="post">
       <tr>
         <td colspan="2">为方便我们及时将书送到您手上,请您确保以下信息真实可靠:
           (带<font color="red"> * </font>号的为必填项,其他的可以不填)</font></td>
       </tr>
       <tr>
         <td colspan="2"><hr size="1" noshade width="100%"></td>
       </tr>
       <tr>
         <td>收货人姓名:</td>
         <td><input name="receiver" type="text" size="20">
           <input name="userid" type="hidden">
         <font color="red"> * </font></td>
       </tr>
       <tr>
         <td>收货人地址:</td>
       <td><input name="address" type="text" size="40">
             <font color="red"> * </font></td>
       </tr>
       <tr>
         <td>联系电话:</td>
           <td><input name="phone" type="text" size="20">
           <font color="red"> * </font></td>
       </tr>
       <tr>
         <td>邮编:</td>
     <td><input name="postcode" type="text" size="20">
           <font color="red"> * </font></td>
```

```
        </tr>
        <tr>
          <td>付费方式：</td>
      <td>
        <select name="payment">
          <option value="邮局汇款">邮局汇款</option>
          <option value="银行转账">银行转账</option>
          <option value="货到付款">货到付款</option>
        </select>
        </tr>
        <tr>
          <td>送货方式：</td>
      <td>
        <select name="deliver">
          <option value="24 小时全国特快专递">24 小时全国特快专递</option>
          <option value="邮局托运">邮局托运</option>
        </select>
      </td>
    </tr>
    <tr>
      <td colspan="2"> <div align="center">
        <input type="submit" value="我要下订单 " onClick="return CheckSubmit();">
        <input type="reset" value="重新填写 ">
        </div></td>
    </tr>
        </form>
      </table>
    </tr>
    <tr>
      <td ><div align="center"><%@include file="bottom.jsp"%></div></td>
    </tr>
</table>
</div>
```

3. 建立服务器端应用程序

order2.jsp 清单如下：

```
<%@ page contentType="text/html; charset=GB2312"%>
<% request.setCharacterEncoding("GB2312"); %>
<link href="maincss.css" rel="stylesheet" type="text/css">
<div align="center">
<table width="750" border="0" cellspacing="1" cellpadding="1">
  <tr>
    <td><div align="center"><%@include file="top.jsp"%></div></td>
  </tr>
```

```
<tr>
  <td><div align="center">
    <table width="100%" border="0" cellpadding="1" cellspacing="1"  class="td">
      <tr>
        <td colspan="6">您已经成功下单,下面是您这次订单信息,
                          我们的工作人员会及时与您联系。</td>
      </tr>
      <tr>
        <td colspan="6"><hr size="1" noshade width="100%"></td>
      </tr>
      <tr>
        <td colspan="2"> 收货人:<%=request.getParameter("receiver")%></td>
      </tr>
      <tr>
        <td colspan="2"> 地址:<%=request.getParameter("address")%></td>
      </tr>
      <tr>
        <td colspan="2"> 邮编:<%=request.getParameter("postcode")%></td>
      </tr>
      <tr>
        <td colspan="2"> 联系电话:<%=request.getParameter("phone")%></td>
      </tr>
      <tr>
        <td colspan="7"> <div align="left">谢谢您的支持!</div></td>
      </tr>
   </table>
</tr>
<tr>
  <td ><div align="center"><%@include file="bottom.jsp"%></div></td>
</tr>
</table>
</div>
```

4. 代码说明

(1) order1.jsp 代码中 <%=request.getParameter("receiver")%>语句的作用是:使用 request 对象接收 form 表单中的 receiver 信息,并把它输出到浏览器中。

(2) order1.jsp 和 order2.jsp 代码中语句:

```
<link href="maincss.css" rel="stylesheet" type="text/css">
```

这两个文件都应用了样式文件"maincss.css",使得页面具有统一风格。

(3) order1.jsp 和 order2.jsp 代码中语句:

```
<%@include file="top.jsp"%>
<%@include file="bottom.jsp"%>
```

使得这两个页面具有相同的顶部和底部。

5. 在浏览器中测试订单处理模块

在浏览器的 URL 栏目中输入 order1.jsp 文件的 URL 地址,显示如图 8-20 所示,如果客户填写正确,单击“我要下订单”运行结果如图 8-21 所示。



图 8-21　订单成功

如果客户填写有缺项,例如缺邮编,系统将提示重新填写,如图 8-22 所示。



图 8-22　输入不正确

习题、上机练习与实训 8

一、习题

1. JSP 提供内置对象的用意是什么? 并列举其 5 种内置对象的功能。
2. 简述 JSP 内置对象 request 和 response 的功能,它们是如何协同工作的?
3. response 对象状态行的作用有哪些?
4. response 对象的 sendRedirect 方法的功能是什么? 常在什么情况下使用?
5. out 对象的功能是什么? 请列举出最常用的两种方法。
6. session 对象的作用是什么? 它在什么范围内共享信息?
7. application 对象的作用是什么? 它在什么范围内共享信息?
8. exception 对象的作用是什么? 它可以增强软件的什么性能?

9. JSP 的 Cookie 对象的作用是什么？是由谁在何处建立的什么文件？

二、上机练习

1. 在购书表单中输入用户名、密码和书名，单击“确定”按钮，将信息发往服务器端，商家根据用户的信息向用户返回应答。

2. 使用 response 对象 sendRedirect 方法，完成第 7 章上机练习 6 的应用。

3. 设计网上考试界面如图 8-23 所示，应用 session 对象存储测试数据，当考生完成试题，单击“确定”按钮，将答案与正确答案比较，给出结果和答题所用的时间。

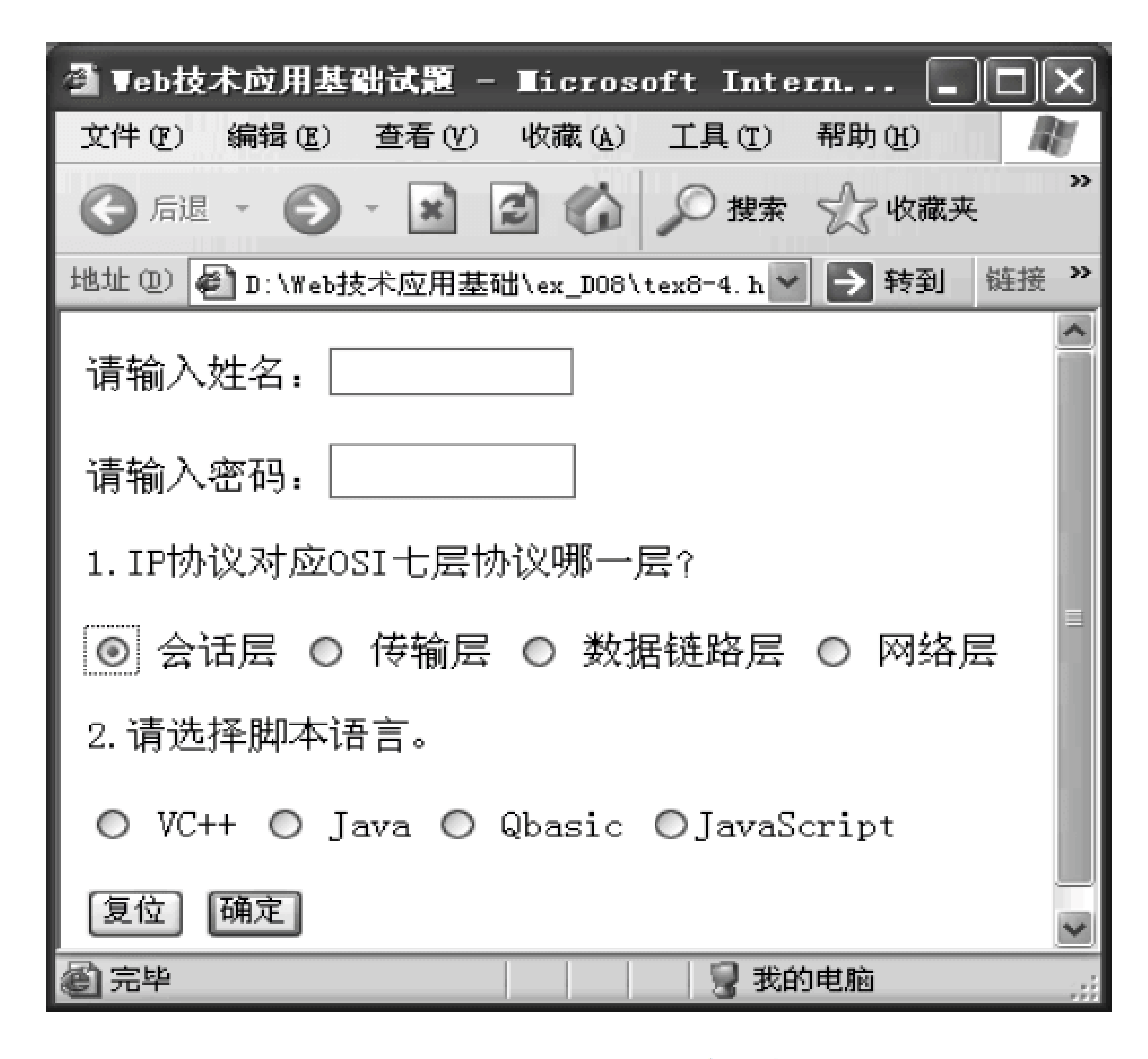


图 8-23　网上考试

4. 模拟 3 个商店(即 3 个页面，供应不同的商品)，应用 session 对象存储用户的姓名、密码和购物信息，当用户购物结束时，向用户返回其购物信息。

5. 应用 application 对象为你的网页制作一个留言板，并统计访客人数。

6. 应用 cookie 对象制作一个登录界面。如果用户没有登录过，显示首次登录的欢迎界面。如果用户已经登录过，则显示上次登录时间并显示欢迎再次登录界面。

7. 应用 session 对象制作分页显示的调查问卷。用户在第一个页面回答需要调查的第一个问题，回答问题后，单击“确定”按钮，页面导向第二个页面。在第二个页面显示第一个问题的回答，并提出第二个问题，用户回答第二个问题后，单击“确定”按钮，导向第三个页面。第三个页面显示用户对前两个问题的回答。

8. 设计用户注册应用，用户在表单中输入需要购买图书的书号，如果输入错误，应用异常处理页面，进行异常处理。

三、实训课题

1. 应用 JSP 技术制作一个如图 8-24 所示的网上购物表单，商家根据用户信息，向用户返回表单的认定信息。

2. 为“Web 技术应用基础”课程制作一个网上考试的应用，要求从应试者登录开始计算时间，交卷后要给出答案、分数和回答问题的时间。

3. 应用 session 对象，完成实训课题 2 的任务。

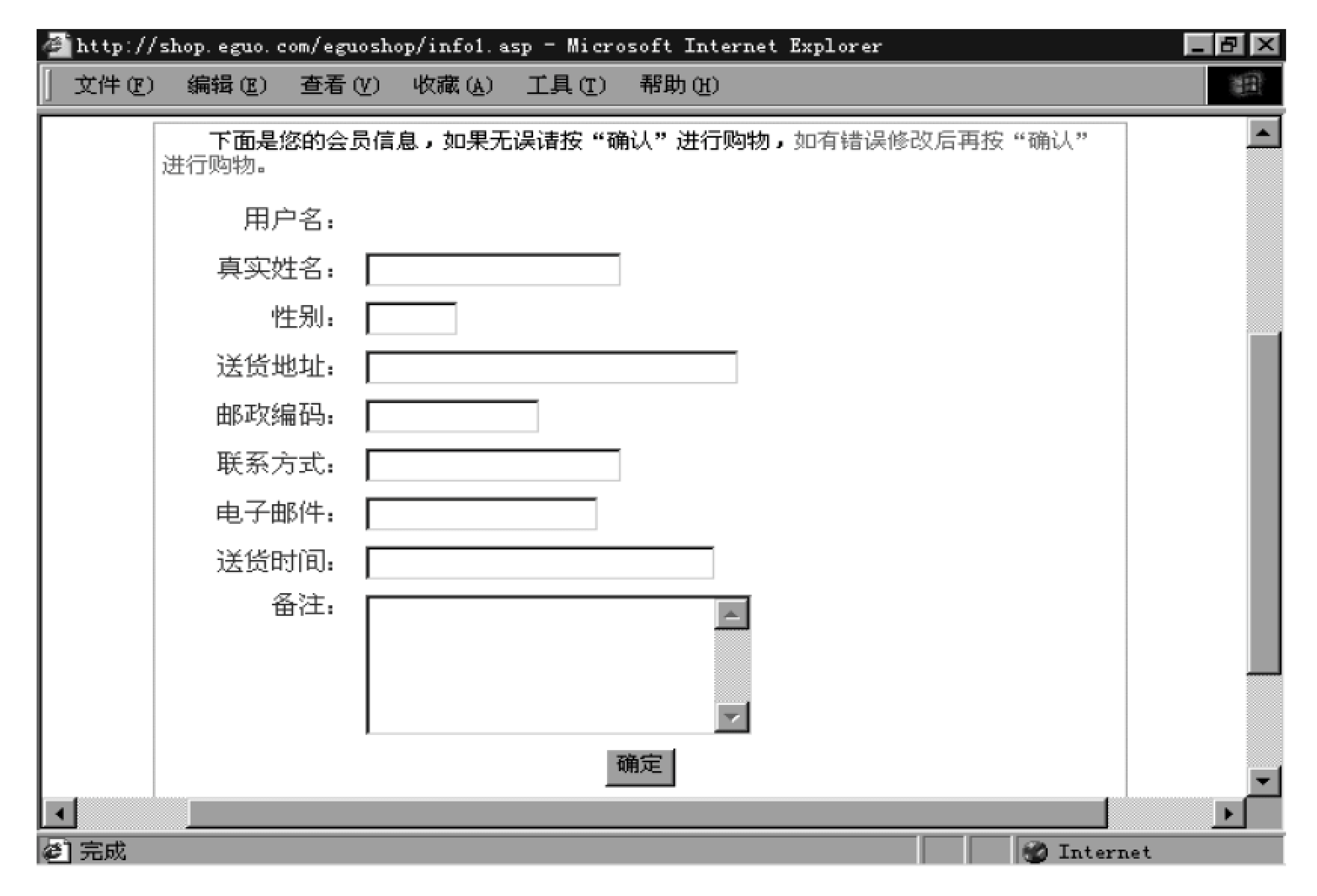


图 8-24　网上购物表单

4. 制作一个计算器，要求计算器比较健壮，能够处理各种异常情况。
5. 使用存取 Cookie 数据的方法，完成实训课题 2 的任务。
6. 为网站完成一个留言板的制作。



第9章 基于JSP的Web数据库应用开发

使用Web方式进行应用系统的开发是当前的一种发展趋势，基于JSP的Web数据库应用开发具有较为广泛的应用。本章将重点介绍JDBC接口技术、数据库连接技术与JSP数据库信息发布技术。

本章案例存放在D：\Tomcat 5.5\webapps\ex_D09目录下。

9.1 Web数据库应用基础

网上应用系统的开发离不开数据库的应用。本节基本上不做理论上的探讨，只对数据库的基础知识和SQL应用做一个简单介绍，力求使事情简单明了，为数据库信息发布做准备。

9.1.1 数据库基本概念

1. 基本术语

(1) 数据库系统

数据库系统是一个存储数据的计算机系统。人们到仓库存取物品，只需走到仓库，请仓库管理员帮他取出所要的物品即可，他并不关心物品在仓库中是如何存放的。要建设一个好的仓库系统，仓库建设者要根据仓库的性质合理地组织仓库中的物品，规划出仓库中要有多少货架，物品如何分类，是否需要内部运输工具等。在合理规划的基础上，实施这个计划，将物品存放到仓库中，并为仓库使用者提供物品存取服务。计算机中的数据库系统与现实世界中的仓库系统类似。数据库系统包括数据仓库的规划、建设及提供货物(数据)的存取服务。数据库系统可以分为数据库和数据库管理系统两部分。

(2) 数据库

数据库是一个按数据结构来存储和管理数据的计算机软件系统。数据库概念包含两层意思：

- 数据库是一个实体，是一个能够合理保管数据的“仓库”，在该“仓库”中存放要管

理的事务数据，“数据”和“库”两个概念结合成为“数据库”。

- 数据库是数据管理的方法和技术，它能够合理地组织数据、方便地维护数据、严密地控制数据和有效地利用数据。

简言之，数据库是一个合理组织了的数据仓库。数据被合理地组织到数据仓库中，使用者可以方便有效地存取数据库中的数据。

(3) 数据库管理系统(DBMS)

数据库管理系统是管理数据库的软件系统，它提供了一组建立数据库和管理数据库的工具。可以使用这些工具进行各种数据库操作，如数据库生成、数据表格生成、数据的输入和修改、数据的检索和使用、数据安全、数据相关关系的设定和数据访问权限的设定等。

目前市场上比较著名的数据库管理系统有 Oracle、Sybase、Informix、SQL Server 等。

2. 数据库设计与管理信息系统

管理信息系统简称为 MIS(Management Information System)，它是计算机应用领域的一个重要分支。管理信息系统帮助人们完成需要手工处理的信息处理工作，不仅能够提高工作效率，降低劳动强度，而且能够提升管理信息的质量和水平。

管理信息系统的核心是数据库。管理信息系统的数据存放在数据库中，数据库技术为管理信息系统提供了数据管理的手段，数据库管理系统为管理信息系统提供了系统设计的方法、工具和环境。管理信息系统、数据库管理系统和数据库的关系如图 9-1 所示。

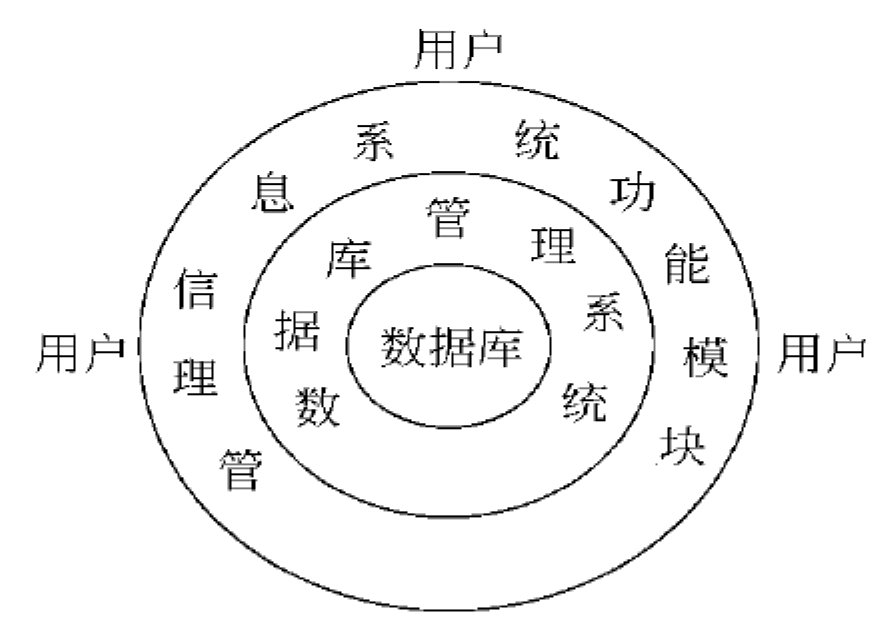


图 9-1　管理信息系统、数据库管理系统和数据库的关系

图 9-1 显示了管理信息系统、数据库管理系统和数据库的关系。管理信息系统的各功能模块通过数据库管理系统提供的工具访问数据库，完成对数据的存取、修改和录入等操作。各种功能模块为用户提供不同的功能要求，并共享数据库中的数据资源，数据库是信息系统的核心。

3. 数据库、表、记录和字段

数据库用数据库名表示。在关系数据库中数据库是多个数据表的集合，通过数据表和表之间的关系定义数据库的结构。例如，网上书店数据库名为 bookshop，库中有 7 个表：book、userinfo、orderform、orderdetail、notes、employee 和 publisher。

表是按某种结构存储的一组数据，它是数据库的基础构件。表类似于日常生活中的表格，如火车时刻表、员工表和电话号码表等。表中数据以行、列方式将相关信息排列成逻辑组。表中每一行称为**记录**，每一列称为**字段**。例如网上书店中的 userinfo 表，见图 9-2。

userid	username	password	gender	address	email	phone	postcode	state
000001	多 特	234567	男	多特教育服务中心	info@dotraining.com	67842111	432200	0
000002	林 琳	123456	女	北京联合大学	linlin@buu.com.cn	64900713	100101	0
000003	彭大双	987654	男	海淀区国兴大厦3楼301室	pds9902@hotmail.com	62535760	100046	1
000004	王 杨	165473	男	北京海淀区学院南路55号	wangyang@sina.com	62257788	100081	1

图 9-2　userinfo 表

9.1.2 创建数据库和表

以网上书店数据库为例说明创建表和数据库的方法。

1. 创建数据库

创建数据库的操作过程如下：

(1) 启动 SQL Server 服务管理器

选择“开始→所有程序→Microsoft SQL Server→服务管理器”，出现如图 9-3 所示的“SQL Server 服务管理器”界面，如果管理器没有启动，单击“开始/继续”按钮启动服务管理器。

(2) 创建数据库

① 选择“开始→所有程序→Microsoft SQL Server→企业管理器”，打开 SQL Server Enterprise Manager 企业管理器。

② 在控制台树中单击“数据库”结点，然后选择“新建数据库”，如图 9-4 所示。

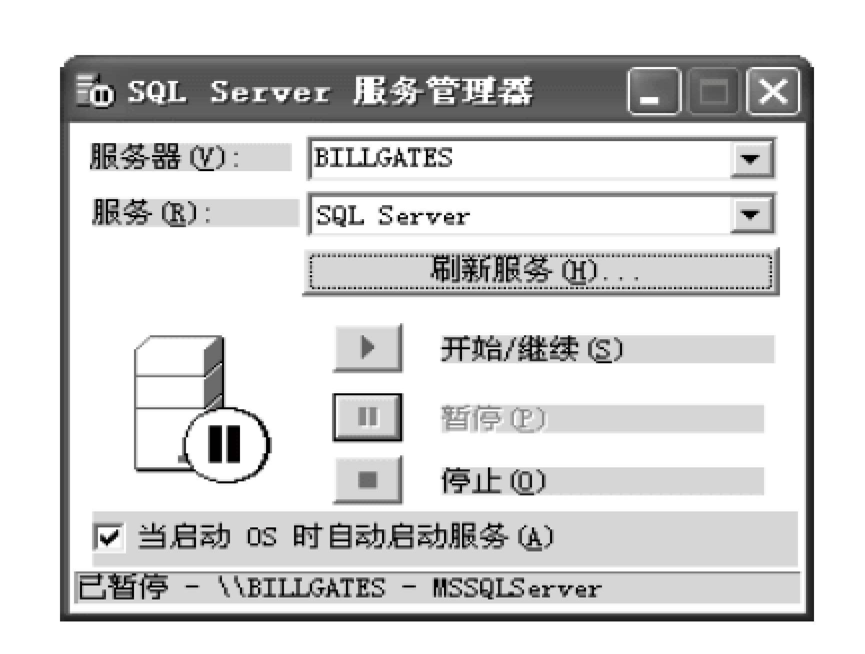


图 9-3 “SQL Server 服务管理器”界面

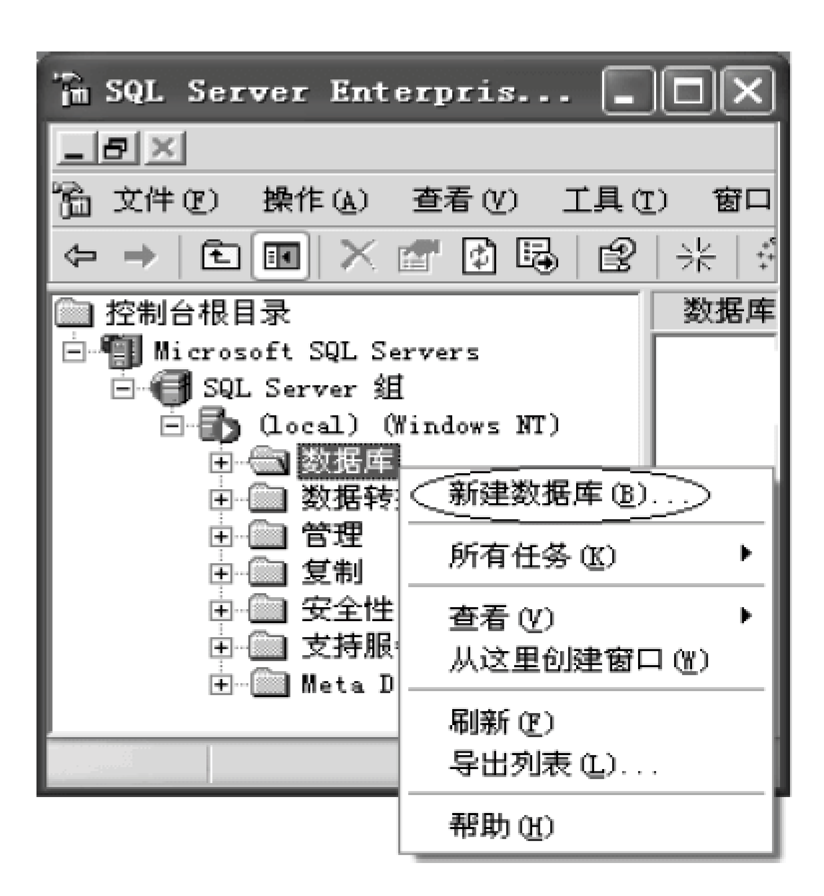


图 9-4 新建数据库

③ 出现“数据库属性”对话框，输入数据库的名称，网上书店数据库的名字定为 bookshop，如图 9-5 所示。

2. 创建新表

为了使读者易于练习，把网上书店实例中的 book 表做了适当的简化，起名为 booktable，以下创建 booktable 表。

① 选择“开始→程序→Microsoft SQL Server→企业管理器→数据库→bookshop”。

② 右击“表”结点，然后选择快捷菜单中的“新建表”命令。

③ 在表设计器窗口设计 booktable 表的结构，输入相应的字段名、类型、长度、是否为空和默认值等。选择 bookid 字段，单击工具栏中的“设置主键”图标，将 bookid 字段设为主关键字。当字段为主关键字时，该字段不允许出现空值。booktable 表的



图 9-5 “数据库属性”对话框

结构如图 9-6 所示。

④ 设计完成后单击工具栏“保存”图标，出现“选择名称”对话框，如图 9-7 所示，在对话框中输入表名 booktable 后，单击“确定”按钮。

列名	数据类型	长度	允许空
bookid	varchar	50	
bookname	varchar	200	✓
author	varchar	50	✓
publisher	varchar	50	✓
pubdate	char	10	✓
price	numeric	9	✓
type	varchar	50	✓
quantity	int	4	✓

图 9-6 booktable 表的结构



图 9-7 输入表名

⑤ 单击标题栏“关闭”按钮。

⑥ 同理制作其他 6 张表，参见第 3 章。

9.1.3 SQL 语句

SQL(Structured Query Language，结构化查询语言)是关系数据库的标准语言，是目前数据库的主流语言。SQL 语言语法完善、功能丰富、综合性强、语句简洁易学，因而备受欢迎。自从 SQL 成为国际标准后，各个数据库厂商纷纷推出各自的支持 SQL 的软件或与 SQL 的接口软件，使得大多数数据库均采用 SQL 作为共同的数据存取语言和标准接口，从而使未来的数据库世界有可能成为一个统一的整体。可以预言，在未来相当长的一段时间内，SQL 将成为数据库语言领域中的一种主流语言。

SQL 具有自含式和嵌入式两种语言形式。自含式 SQL 能够独立地进行联机交互，在终端键盘上直接键入命令就可以执行 SQL 语句，对数据库进行操作；嵌入式 SQL 能够

嵌入到其他编程语言(如 C、Visual C++ 、Visual Basic、Delphi 和 PowerBuilder 等)及脚本语言(如 VBScript 和 JavaScript 等)中实现对数据库的操作。

1. SQL 主要功能

SQL 的功能可分为三类。

(1) 数据定义功能

SQL 的数据定义功能通过数据定义语言(Data Definition Language,DDL)实现,它用来定义关系数据库的模式、内模式和外模式,以实现对基本表、视图及索引文件的定义。

(2) 数据操纵功能

SQL 的数据操纵功能通过数据操纵语言(Data Manipulation Language,DML)实现。DML 包括数据查询和数据更新两种数据操作语句。数据查询是指对数据库中的数据查询、统计、分组、排序、检索等功能,数据更新是指数据的插入、删除和修改等数据的维护操作。

(3) 数据控制功能

SQL 的数据控制功能通过数据控制语言(Data Control Language,DCL)实现。数据控制是指数据的安全性和完整性的控制。SQL 通过对数据库用户的授权和收权命令来实现有关数据的存取控制,以保证数据库的安全性。SQL 还提供了数据完整约束条件的定义和检查机制,以保证数据库的完整性。

2. SQL 的构成

SQL 由命令、子句和运算符等元素构成,这些元素结合起来组成用于创建、更新和操作数据库的语句。

(1) 数据定义语句

数据定义语句(DDL)用来定义新的数据库、表、字段和索引。

- CREATE :创建新的表、字段和索引。
- DROP:删除数据库中的表和索引。
- ALTER:通过添加字段或改变字段定义来修改表。

(2) 数据操纵语句

数据操纵语句(DML)命令用来创建查询,以便从数据库中筛选、抽取和排序数据。

- SELECT:在数据库中查找满足特定条件的记录。
- INSERT:在数据库中插入新的记录。
- UPDATA:更新数据库中的记录和字段。
- DELETE:从数据库中删除记录。

(3) 数据控制语句

数据控制语句(DCL)命令通过对数据库用户使用权限的限制来保证数据的安全性。

- GRANT:为数据库用户授予某种权限。
- REVOVE:收回数据库用户的某种权限。
- DENY:拒绝用户访问数据库的某一对象。

（4）SQL 子句

SQL 子句用于定义要选择或操作的数据。

- FROM：指定要操作的表。
- WHERE：指定选择记录时要满足的条件。
- GROUP BY：将选择的记录分组。
- HAVING：指定分组的条件。
- ORDER BY：按特定的顺序排序记录。

（5）SQL 运算符和计算函数

SQL 运算符分为两类：逻辑运算符和比较运算符。逻辑运算符（AND、OR 和 NOT）用于连接两个表达式，通常在 WHERE 子句中使用。比较运算符（＜、＜＝、＞、＞＝、＝、＜＞、BETWEEN、LIKE 和 IN）用于比较两个表达式的值。

SQL 有一些计算函数，例如，用 AVG 函数计算平均值，用 COUNT 函数计算返回记录数。

3. 创建表（CREATE TABLE）

使用 CREATE TABLE 语句在数据库中创建表，同时建立相关字段及其数据类型，创建表的基本语法规则如下：

```
CREATE TABLE [库名] 表名(列名 数据类型 [(字符串长度)]… [,…n])
```

其中“[]”是可选项，[,…n]表示可以重复前面的内容，定义多个字段。

4. 选择语句（Select Statement）

选择语句的功能是从现有的数据库中检索数据，将满足一定约束条件的一个或多个表中的字段从数据库中挑选出来，并按一定的分组和排序方法显示。简单的选择语句只有 FROM 子句，用来指定数据的来源，也就是来自哪些表。如果某一字段出现在多个表中，要用“.”运算符指定字段所属的表。WHERE 子句指定选择记录时要满足的条件，如果没有 WHERE 子句，则返回表中的所有记录。选择语句的语法规则如下：

```
SELECT 字段 1[,字段 2,…]FROM 表名[WHERE 条件表达式]
```

例如：

① 在 booktable 表中检索所有书名，SQL 语句格式是：

```
SELECT   bookname FROM booktable
```

该语句的功能是：从 booktable 表的 bookname 字段检索数据。它们的语义如下。

SELECT：关键字，说明要做查询操作。

bookname：字段名，数据从该字段中检索。

FROM：关键字，指明信息来源，后面列出表的名称。

booktable：表的名称，数据从该表中检索。

结果集是 booktable 表中的全部书名。

② 在 booktable 表中查询书号等于 ISBN 7-302-08599-4 的书名,SQL 语句格式是:

```
SELECT  bookname FROM booktable WHERE bookid=ISBN 7-302-08599-4
```

结果集是书号等于 ISBN 7-302-08599-4 的图书。

③ 范围查询,在 booktable 表中查询 2008 年上半年出版的书的书名,SQL 语句格式是:

```
SELECT bookname FROM booktable WHERE pubdate Between '2008-1-1'and '2008-6-30'
```

④ 在选择语句中增加 ORDER BY 子句,可以使结果集按序排列。例如,将图书按出版日期进行排序,SELECT 语句格式是:

```
SELECT * FROM booktable  ORDER BY pubdate
```

默认值是升序,如果需要按降序排列,在要排序的字段后加关键字 DESC,格式如下:

```
SELECT * FROM booktable ORDER BY pubdate DESC
```

⑤ 模糊查询,在 WHERE 子句中应用 LIKE 运算符,可以只选择与用户规定格式相同的记录。用通配符"%"可以代替任何字符串。例如,在 userinfo 表中查询所有姓王的读者,可用以下语句:

```
SELECT * FROM userinfo WHERE username LIKE '王%'
```

5. 插入语句(Insert Statement)

插入语句的功能是把一个或多个记录添加到指定表中。插入语句的语法规则如下:

INSERT INTO 表名(字段 1[,字段 2,…,n])VALUES(值 1[,值 2,…,n])

例如,在 booktable 表中插入新书的记录(书号:ISBN 7-04-012301-0,书名:C++ 程序设计,作者:吴乃陵,出版社编号:3,出版日期:2003-8-1,定价:29.5,类别:计算机,数量:100),插入语句如下:

```
INSERT INTO booktable (bookid, bookname, author, publisher, pubdate, price,type, quantity)
  VALUES('ISBN 7-04-012301-0', 'C++ 程序设计', '吴乃陵', '高等教育出版社', '2003-8-1',29.5,
   '计算机',100)
```

6. 更新语句(Update Statements)

更新语句按某个条件来更新表中的字段。更新语句的语法规则如下:

UPDATE 表名 SET 列名=表达式[,列名=表达式,…][WHERE 条件表达式]

例如,在 booktable 表中把所有图书的在库册数减 2,语句如下:

```
UPDATE booktable SET quantity=quantity-2
```

7. 删除语句(Delete Statement)

DELETE 语句的功能是删除由 FROM 子句列出的、满足 WHERE 子句条件的一个或多个表中的记录。语法规则如下:

```
DELETE FROM 表名 [WHERE 条件表达式]
```

例如,在 booktable 表中删除刚才插入的书号为 ISBN 7-04-012301-0 的记录,语句如下:

```
DELETE FROM booktable WHERE bookid=ISBN 7-04-012301-0
```

又如:下条 SQL 语句的功能将删除表中所有的记录,使用时要特别当心:

```
DELETE FROM booktable
```

9.2 JDBC 接口技术

9.2.1 JDBC 概述

1. 什么是 JDBC

JDBC(Java Database Connectivity)接口技术实际上是一种通过 Java 语言访问数据库的应用程序接口(API)。许多数据库系统带有 JDBC 驱动程序,Java 程序通过 JDBC 驱动程序与数据库连接,执行查询、插入、更改和删除等操作。为能够访问由 ODBC 驱动程序接口的数据库,Sun 公司开发了 JDBC-ODBC Bridge,应用这项技术,Java 程序就能够访问由 ODBC 驱动的数据库。由于大多数数据库系统都带有 ODBC 驱动程序,所以使用 JDBC-ODBC Bridge 技术 Java 程序可以访问大多数数据库,如:MS SQL Server、Oracle、Sybase、Informix 和 MS Access 等。

2. JDBC 的功能

JDBC 的主要功能如下:

① 与一个数据库建立连接(connection)。

② 向数据库发送 SQL 语句(statement)。

③ 处理数据库返回的结果(resultset)。

3. JDBC 访问数据库的 4 种方式

JDBC 对数据库的访问可有 4 种方式。

(1) JDBC-ODBC 桥驱动程序

JDBC-ODBC 桥利用 ODBC 驱动程序提供 JDBC 访问。JDBC-ODBC 桥是比较通用

的数据库接口，它利用了微软的 ODBC 的开放性，只要本地机装有 ODBC 驱动，采用 JDBC-ODBC 桥驱动几乎可以访问所有的数据库。所以对于已经安有 ODBC 驱动的客户端，这种方式是可行的。

（2）本地 API 驱动

本地 API 驱动直接把 JDBC 调用转变成数据库标准调用，然后再去访问数据库。这种方法也需要本地数据库驱动程序。

（3）网络协议驱动

使用一段纯 Java 代码，把 JDBC 调用转换成目标数据库网络协议调用。

（4）本地协议驱动

使用一段纯 Java 代码，把 JDBC 调用转换成数据库本地协议调用。

9.2.2 JDBC-ODBC 桥

使用 JDBC-ODBC 桥连接访问数据库，先要建立数据源（Data Source Name，DSN），这个数据源对应一个数据库。为了连接到数据库，需要建立一个 JDBC-ODBC 桥接器，也就是加载 JDBC-ODBC 桥驱动程序。

以网上书店的 bookshop 数据库为例，说明如何利用 JDBC-ODBC 桥建立数据库连接。

一个数据源就是对数据库的一个命名连接。数据源有 3 种：用户数据源、系统数据源和文件数据源。用户数据源只有用户可以看见，只能用于当前机器中。系统数据源是允许任何具有权限的用户都可以访问的数据源。文件数据源把信息存储在后缀为.dsn 的文本文件中，如果把该文件放在网络共享目录中，则可被网络中任何一台工作站访问到。Web 应用程序访问数据库时，通常是建立系统数据源。

以下为建立数据源的操作步骤，访问的数据库类型为 SQL Server，连接的数据库名是 bookshop，数据源名是 bookshoplk，用户名为 sa，密码无。操作步骤如下：

① 打开“控制面板→管理工具→数据源(ODBC)”，选择“系统 DSN”选项卡，如图 9-8 所示。

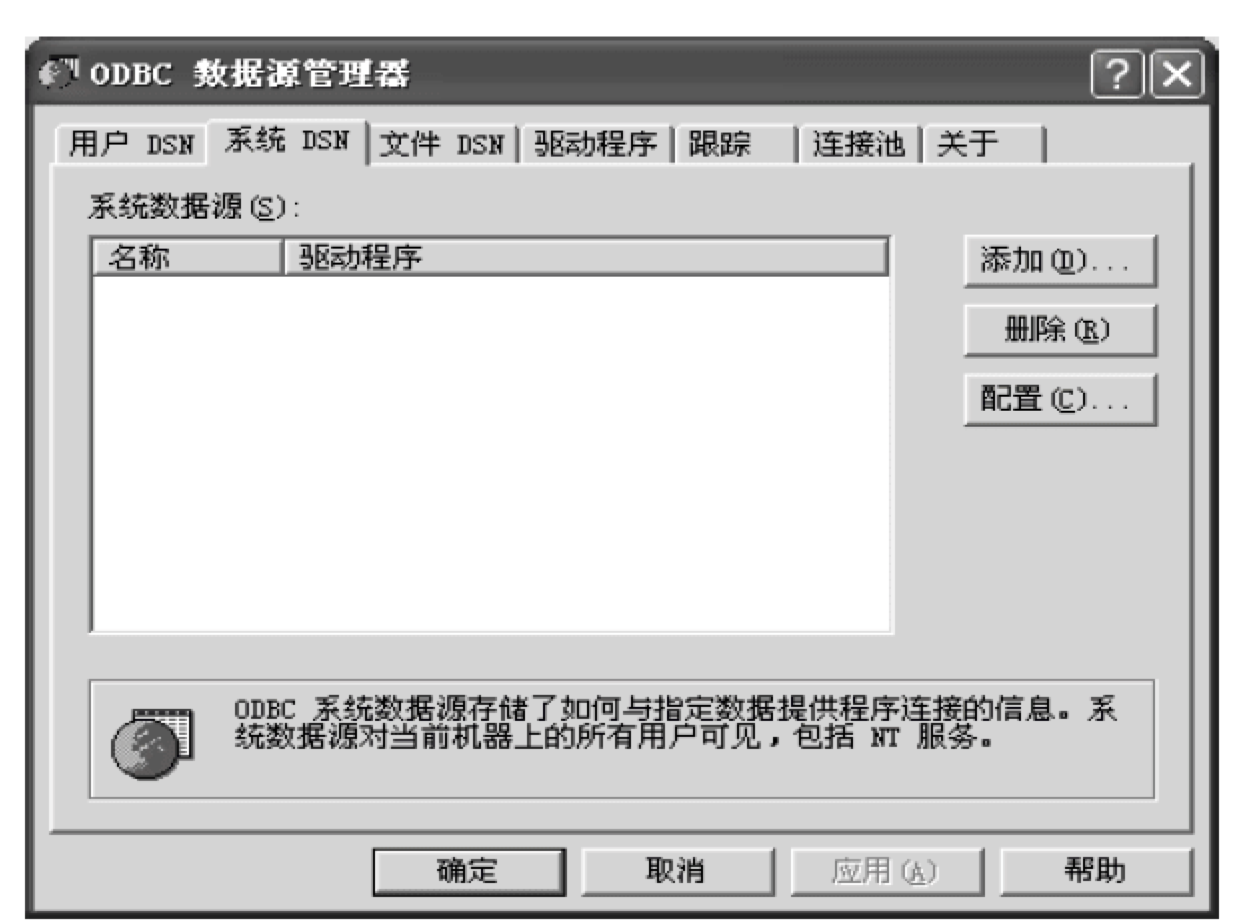


图 9-8 “ODBC 数据源管理器”对话框的“系统 DSN”选项卡

② 单击“添加”按钮，弹出“创建新数据源”对话框，如图 9-9 所示。



图 9-9 “创建数据源”对话框

③ 选择 SQL Server，单击“完成”按钮。

④ 弹出“创建到 SQL Server 的新数据源”窗口，在名称文本框输入数据源名称，在网上书店中，为数据源起名为 bookshoplk。在服务器文本框输入(local)。单击“下一步”按钮，如图 9-10 所示。

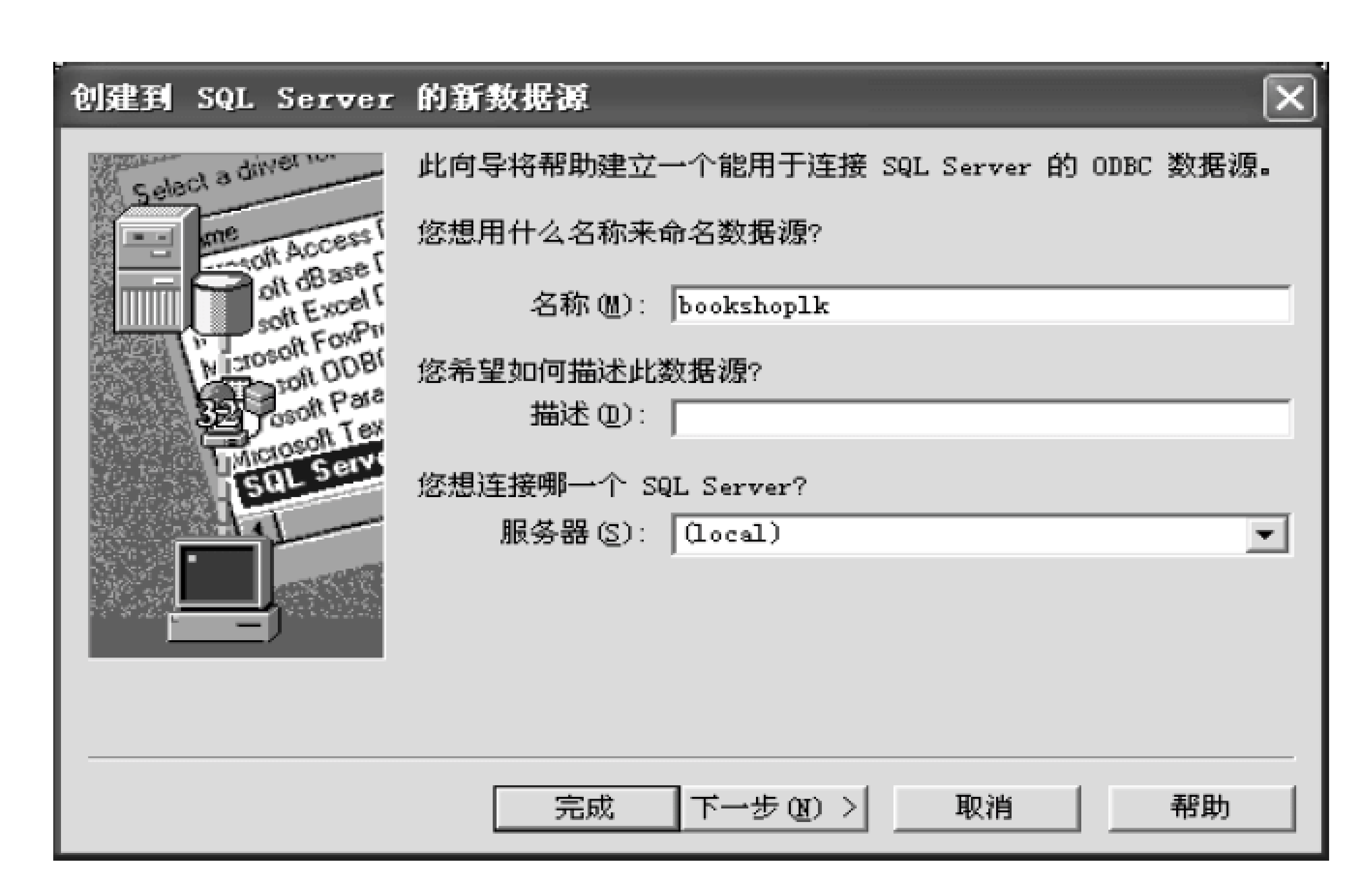


图 9-10 “创建到 SQL Server 的新数据源”对话框

⑤ 在弹出的窗口中，选择“使用用户输入登录 ID 和密码的 SQL Server 验证”单选按钮，在“登录 ID”和“密码”文本框中输入对 bookshop 数据库有存取权限的 SQL Server 账号和密码。(用户也可以不选“使用用户输入登录 ID 和密码的 SQL Server 验证”。)单击“下一步”按钮，如图 9-11 所示。

⑥ 在弹出的窗口中，选择“更改默认的数据库为”复选框，并选择 bookshop 数据库，单击“下一步”按钮，如图 9-12 所示。

⑦ 在弹出的窗口中单击“完成”按钮。

⑧ 在弹出的“ODBC Microsoft SQL Server 安装”窗口中单击“测试数据源”按钮，如图 9-13 所示。

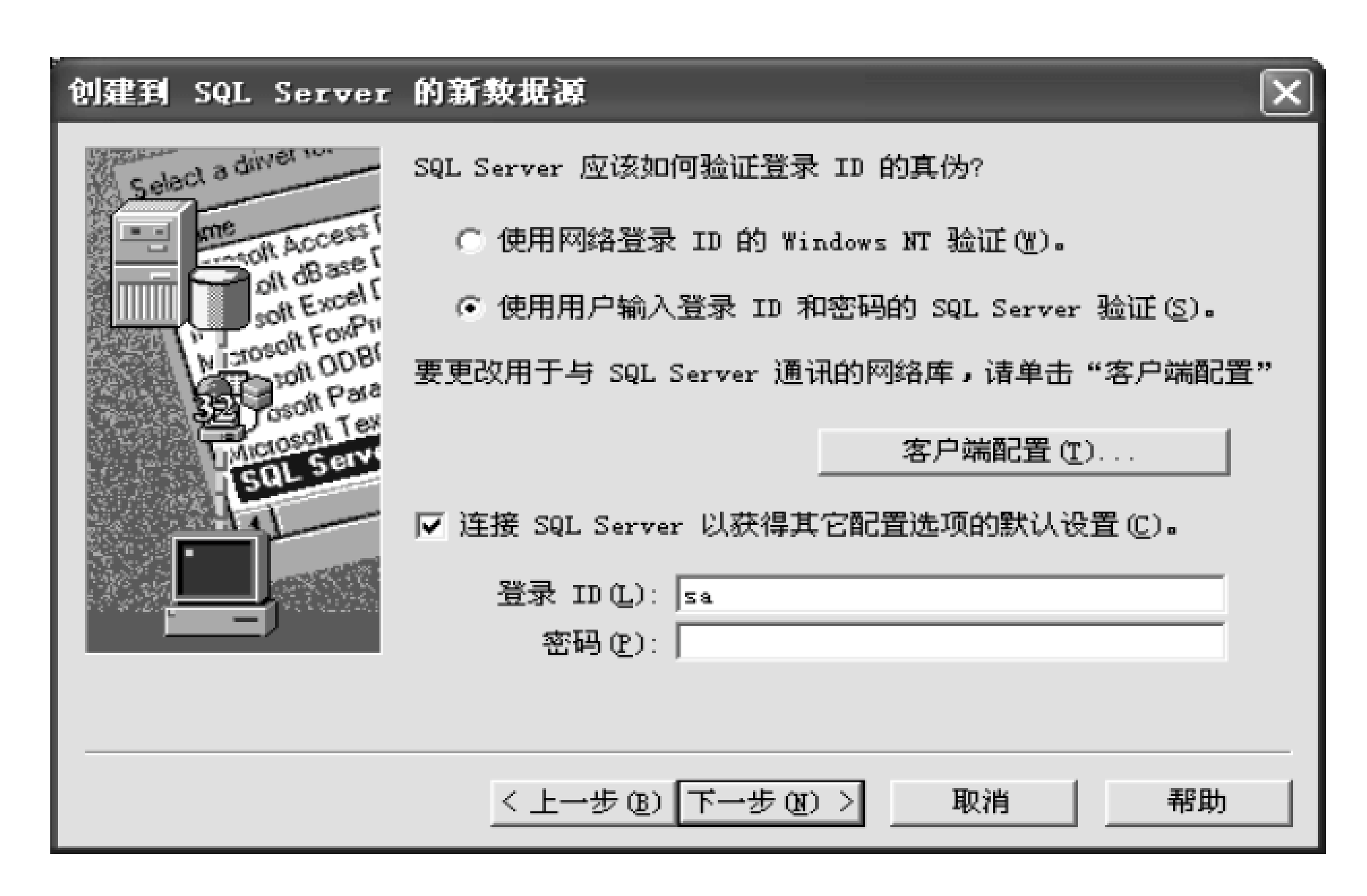


图 9-11　选择 SQL Server 验证登录 ID 方式



图 9-12　将数据源对应的默认数据库改为 bookshop

⑨ 弹出“SQL Server ODBC 数据源测试”窗口，如果对话框提示测试成功，表示 DSN 设置正确，单击“确定”按钮，完成系统 DSN 的建立，如图 9-14 所示。



图 9-13　“ODBC Microsoft SQL Server 安装”窗口



图 9-14　测试成功

9.2.3 JDBC 建立数据库连接示例

使用 JDBC 建立数据库连接的过程比较繁杂，本书先举例说明连接是如何进行的，然后总结出它的几个工作过程，最后详细讲解各个过程的工作原理和细节。

本书使用 JDBC-ODBC 桥驱动程序建立与数据库的连接。

例 9.1 使用 JSP 技术查询 booktable 表（见图 9-15）中书号是 ISBN 7-04-012301-0 的图书。本例使用上文生成的 bookshoplk 数据源，通过 JDBC-ODBC 桥访问数据库。

表“booktable”中的数据，位置是“bookshop”中、“(local)”上

bookid	bookname	author	publisher	pubdate	price	type	quanti
ISBN 7-04-012301-0	C++程序设计	吴乃陵	高等教育出版社	2003-8-1	29.5	计算机	100
ISBN 7-302-08599-4	C++程序设计	谭浩强	清华大学出版社	2004-6-1	36	计算机	100
ISBN 7-35789-925-256	Google搜索从入门到精通	网友	电子工业出版社	2004-7-27	25	计算机	150
ISBN 7-5053-9856-3	ASP&ASP.NET应用编程150例	王兴东	电子工业出版社	2004-5-1	35	计算机	200
ISBN 7-5606-0939-2	Internet/Intranet应用与网页制作实训	樊月华	西安电子科技大学出版社	2003-1-1	15	计算机	100
ISBN 7-5640-0139-9	CPA会计	武玉荣	电子工业出版社	2003-3-1	30	经济	100
ISBN 7-5640-0165-8	现代汽车动力传动装置的控制技术	林学东	清华大学出版社	2004-7-27	30	机械	150
ISBN 7-5640-0244-1	火星的故事	[英]帕特里克.摩尔	人民邮电出版社	2004-5-1	15	小说	100
ISBN 978-7-04-021459-8	计算机应用基础	刘艳丽	高等教育出版社	2007-6-1	29.5	计算机	100
ISBN 978-7-121-05285	计算机网络工程与实训教程	徐武	电子工业出版社	2008-1-1	28	计算机	100
ISBN 978-7-302-12927-4	Web程序设计	(美)Robert W.Sebesta	清华大学出版社	2006-8-1	58	计算机	100
ISBN 978-7-302-15261-3	Web技术	Jeffrey C.Jackson	清华大学出版社	2007-6-1	59	计算机	100
ISBN 978-7-900222-91-6	计算机网络	谢希仁	电子工业出版社	2008-2-1	29	计算机	200

图 9-15 booktable 表

代码 ex9-01.jsp 清单如下：

```
<%@ page contentType="text/html; charset=GB2312"%>
<%@ page import="java.sql.*"%>
<html>
<head><title>JDBC 建立数据库连接</title>
</head>
<body><center>
<font size=5 color=blue>数据查询</font><hr>
<%
  //加载驱动程序
  Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
  //建立连接
  Connection conn=DriverManager.getConnection("jdbc:odbc:bookshoplk","sa","");
  //发送 SQL 语句
  Statement stmt=conn.createStatement();
  try{
  //建立 ResultSet(结果集)对象
  ResultSet rs;
  //执行 SQL 语句
  rs=stmt.executeQuery("SELECT * FROM booktable where bookid='ISBN 7-04-012301-0'");
%>
<table border=3>
```

```
  <tr bgcolor=silver><b>
     <td>bookid</td><td>bookname</td><td>author</td><td>publisher</td>
     <td>pubdate</td><td>price</td><td>type</td><td>quantity</td>
</b></tr>
<%
   //利用 while 循环将数据表中的记录列出
  while (rs.next()){
%>
  <tr>
       <td> <%=rs.getString("bookid")%></td>
       <td> <%=rs.getString("bookname")%></td>
       <td> <%=rs.getString("author")%></td>
       <td> <%=rs.getString("publisher")%></td>
       <td> <%=rs.getString("pubdate")%></td>
       <td> <%=rs.getString("price")%></td>
       <td> <%=rs.getString("type")%></td>
       <td> <%=rs.getString("quantity")%></td>
  </tr>
<%
  }
  rs.close();              //关闭 ResultSet 对象
  }
  catch(Exception e){
  out.println(e.getMessage());
  }
  stmt.close();            //关闭 Statement 对象
  conn.close();            //关闭 Connection 对象
%>
</table></center>
</body></html>
```

代码 ex9-01.jsp 的运行结果如图 9-16 所示。



图 9-16　JDBC 建立数据库连接

9.2.4 JDBC 建立数据库连接方法详解

1. JDBC 建立数据库连接步骤

通过例 9.1 可以得出,JDBC 建立数据库连接需要经过以下几个步骤:

① 加入命令行:

```
<%@ page import="java.sql.*"%>
```

② 加载驱动程序:

```
Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
```

③ 建立连接:

```
Connection conn=DriverManager.getConnection("jdbc:odbc:bookshoplk","sa","");
```

④ 发送 SQL 语句:

```
Statement stmt=conn.createStatement();
```

⑤ 建立 ResultSet(结果集)对象:

```
ResultSet rs;
```

⑥ 执行 SQL 语句:

```
rs=stmt.executeQuery("SELECT * FROM booktable where bookid='ISBN 7-04-012301-0'");
```

⑦ 关闭对象:

```
rs.close();             //关闭 ResultSet 对象
stmt.close();           //关闭 Statement 对象
conn.close();           //关闭 Connection 对象
```

2. JDBC 数据库连接步骤详解

(1) 加入命令行

所有与数据库有关的对象和方法都在 java.sql 包中,所以在使用 JSP 访问数据库的程序中必须加入命令行:

```
<%@ page import="java.sql.*"%>
```

(2) 加载驱动程序

使用 JDBC-ODBC 桥方式连接数据库,必须先加载 JDBC-ODBC 桥驱动程序,语句如下:

```
Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
```

Class 是包 java.lang 中的一个类,该类通过调用静态方法 forName 建立 JDBC-ODBC 桥接器,即加载驱动程序。

由于加载驱动程序时可能产生异常，所以需要异常处理程序段。

```
try{ …
}
catch(Exception e ){
    …
}
```

(3) 建立连接

要连接一个数据库，必须创建 Connection 类的一个实例，语法规则如下：

Connection conn=DriverManager. getConnection(jdbc:odbc:数据源名,"Loginname","Password");

调用 DriverManager. getConnection 方法建立与数据库的连接，该方法指定了数据库的位置、用户名和用户口令。DriverManager 类位于 JDBC 的管理层，作用在用户和驱动程序之间。它跟踪可用的驱动程序，并在数据库和相应的驱动程序之间建立连接。数据源名是在"控制面板→管理工具→数据源(ODBC)"中设置的数据源名。"Loginname"是用户名，"Password"是用户口令。如果在数据源中没有设置用户名和用户口令，连接形式如下：

```
Connection conn=DriverManager. getConnection(jdbc:odbc:数据源名,"","");
```

一旦 DriverManager. getConnection 方法找到了建立连接的驱动程序和数据源，则通过用户名和口令开始与 DBMS 建立连接，如果连接通过，连接建立完成。

(4) 发送 SQL 语句

JDBC 提供了 3 个类向数据库发送 SQL 语句：Statement、PrepareStatement 和 CallableStatement。

• Statement 类的对象由 Connection 的 createStatement 方法创建，用于发送不带参数的简单 SQL 语句，对数据库进行具体操作，如查询、修改等。在执行一个 SQL 查询语句前，必须用 createStatement 方法建立一个 Statement 类的对象。例如：

```
Connection conn=DriverManager. getConnection("jdbc:odbc:bookshoplk","sa","");
Statement stmt=conn. createStatement();
```

• PrepareStatement 类的对象由 Connection 的 PrepareStatement 方法创建，它用来执行带或不带 IN 参数的预编译 SQL 语句。Statement 对象在每次执行 SQL 语句时都将该语句发送给数据库，所以执行效率比较低。而 PrepareStatement 对象预编译过，所以执行速度比 Statement 快。因而多次执行的 SQL 语句常被创建成 PrepareStatement 对象。例如：

```
Connection conn=DriverManager. getConnection("jdbc:odbc:bookshoplk","sa","");
PrepareStatement pstmt=conn. PrepareStatement("SELECT * FROM booktable");
```

• CallableStatement 类的对象由 Connection 的 PrepareCall 方法创建，它用来执行数据库已存储过程的调用。例如：

```
Connection conn=DriverManager.getConnection("jdbc:odbc:bookshoplk","sa","");
CallableStatement cstmt=conn.PrepareCall ("{call getData(?,?)}");
```

其中,getData 是存储过的过程名,“?”表示:IN、OUT 或 INOUT。

(5) 创建结果集对象

一旦连接到数据库,就可以查询数据表名、列名和有关的信息,并且可以运行 SQL 语句对数据库中的数据进行查询、添加、更新和删除等操作。JDBC 提供了 ResultSet、DatabaseMetaData 和 ResultSetMetaData 类获取数据库中的信息。

ResultSet 类存放查询结果,并通过一套方法提供对数据的访问。它是 JDBC 中很重要的对象。ResultSet 包含任意数量的命名列,可以按名字访问这些列;它也包含一或多个行,可以按顺序自上而下地逐一访问。例如:

```
Statement stmt=con.createStatement();
ResultSet rs;
rs=stmt.executeQuery("SELECT * FROM booktable where bookid='ISBN 7-04-012301-0'");
```

当建立一个 ResultSet 类对象时,它指向第一行之前的位置。ResultSet 对象常用方法如下。

- getInt(int):将序号为 int 的列的内容作为整数返回。
- getInt(String):将名称为 String 的列的内容作为整数返回。
- getFloat(int):将序号为 int 的列的内容作为一个 float 型数返回。
- getFloat (String):将名称为 String 的列的内容作为 float 型数返回。
- getData(int):将序号为 int 的列的内容作为日期返回。
- getData(String):将名称为 String 的列的内容作为日期返回。
- next():把行指针移到下一行,如果没有剩余行,则返回 false。
- close():关闭结果集。
- getMetaData():返回 ResultSetMetaData 对象。

ResultSetMetaData 类实例提供 ResultSet 中列的名称、数目和类型信息。例如:

```
ResultSetMetaData rsmd;
rsmd=Results.getMetaData();
NumCols=rsmd.getColumnCount();
```

ResultSetMetaData 对象常用方法如下。

- getColumnCount():返回 ResultSet 中的列数。
- getColumnName(int):返回序号为 int 的列名。
- getColumnLabel(int):返回序号为 int 列暗含的标签。
- isCurrency(int):如果此列包含有货币单位的一个数字,则返回 true。
- isReadOnly(int):如果此列是只读,则返回 true。
- isAutoIncrement(int):如果此列自动递增,则返回 true。

DatabaseMetaData 类实例提供整个数据库的信息,如:表名、表的索引、数据库产品名和版本、数据库支持的操作等。

例 9.2 代码 ex9-02.jsp 输出 booktable 表中各列的名称。代码创建了 ResultSetMetaData 对象 rsmd，并使用 getColumnCount() 和 getColumnName() 方法取得 booktable 表中的列数和列名。代码清单如下：

```
<%@ page contentType="text/html; charset=GB2312"%>
<%@ page import="java.sql.*"%>
<html>
<head><title>输出 booktable 表各列的名称 </title>
</head>
<body><center>
<font size=4 color=blue>输出 booktable 表各列的名称</font><hr><br>
<%
  Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
  //建立连接
  Connection conn=DriverManager.getConnection("jdbc:odbc:bookshoplk","sa","");
  //发送 SQL 语句
  Statement stmt=conn.createStatement();
  ResultSet rs=stmt.executeQuery("SELECT * FROM booktable");
//建立 ResultSet 对象，并执行 SQL 语句
  ResultSetMetaData rsmd=rs.getMetaData(); //创建 ResultSetMetaData 对象
%><p>
记录集中共有<font size=4 color=red>
<%=rsmd.getColumnCount() %></font>列，各列的名称是：<br>
<font size=4 color=red>
<%
for(int i=1; i <=rsmd.getColumnCount(); i++){
    if(i==1)
        out.print(rsmd.getColumnName(i));
    else
        out.print(", "+rsmd.getColumnName(i));
}
rs.close();      //关闭 ResultSet 对象
stmt.close();  //关闭 Statement 对象
conn.close();  //关闭数据库连接对象
%></font>
</body></html>
```

代码 ex9-02.jsp 的运行结果如图 9-17 所示。

(6) 执行 SQL 语句

Statement 对象提供了 3 种执行 SQL 语句的方法：executeQuery、executeUpdate 和 execute。

- executeQuery：用于产生单个结果集的语句，例如 select 语句：

```
rs=stmt.executeQuery("SELECT * FROM booktable");
```

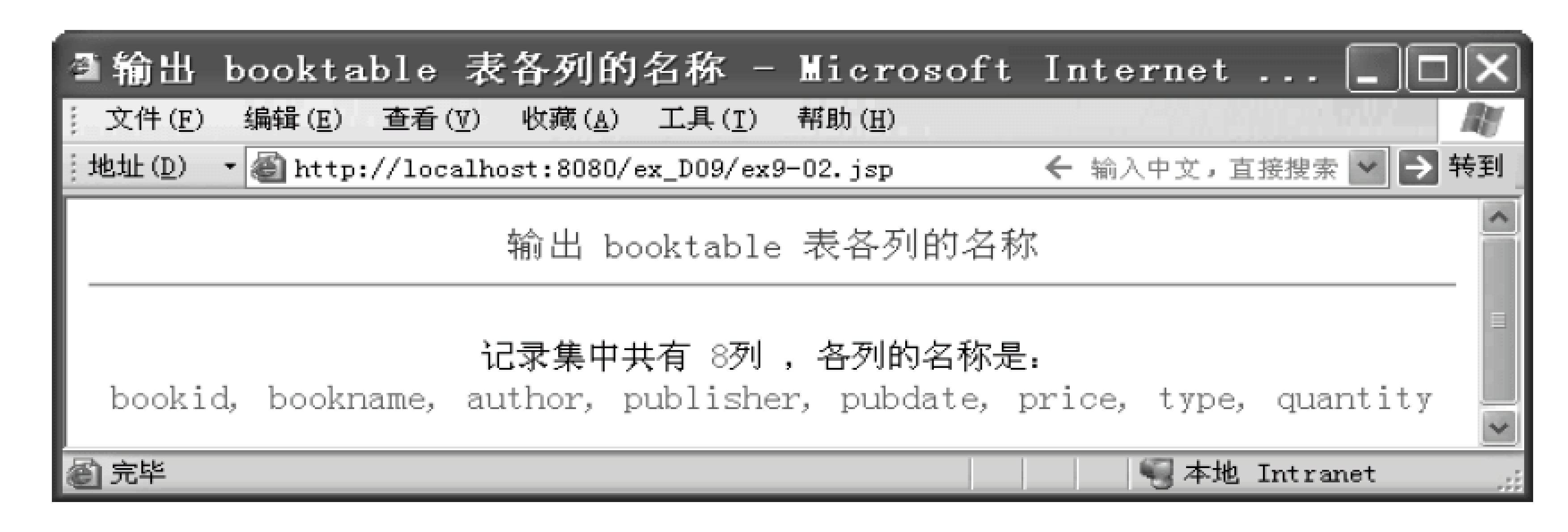


图 9-17　输出 booktable 表各列的名称

- executeUpdate：用来执行 insert、update、delete 以及 sqlddl(数据定义语句)。
- execute：用来返回多个结果集、多个更新计数或两者组合的语句。

本章以下几节仍然以 booktable 数据表为例，学习对数据库信息的选择、插入、删除和更新操作。

9.3　查询记录

9.3.1　顺序查询

使用 ResultSet(结果集)的 next()方法顺序输出一个表里的所有记录。

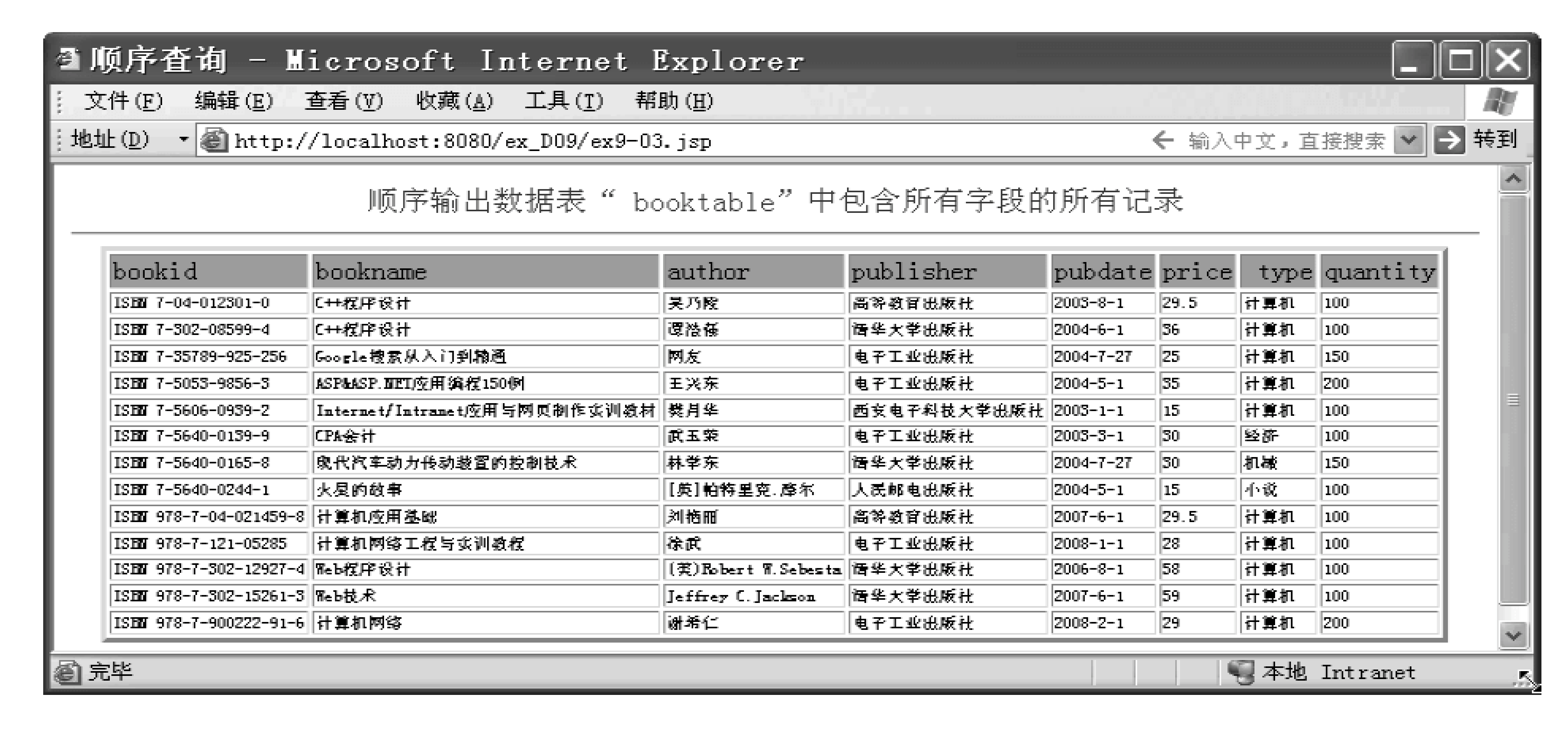
例 9.3　顺序输出数据表“booktable”中的所有记录和所有字段。使用代码："Select * From booktable"从booktable 数据表中选择所有的记录，放置在 rs 结果集中，然后使用 rs.next()方法将结果集中的数据顺序显示出来。

ex9-03.jsp 代码清单如下：

```
<%@ page contentType="text/html; charset=GB2312"%>
<%@ page import="java.sql.*"%>
<html>
<head><title>顺序查询</title>
</head>
<body><center>
<font size=4 color=blue>顺序输出数据表“booktable”中包含所有字段的所有记录</font><hr>
<%
  Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
  Connection conn=DriverManager.getConnection("jdbc:odbc:bookshoplk","sa","");
  Statement stmt=conn.createStatement();
  try{
  ResultSet rs; //建立 ResultSet(结果集)对象
  rs=stmt.executeQuery("SELECT * FROM booktable"); //执行 SQL 语句
%>
<table border=3>
  <tr bgcolor=silver><b>
```

```
    <td>bookid</td><td>bookname</td><td>author</td><td>publisher</td>
    <td>pubdate</td><td>price</td><td>&nbsptype</td><td>quantity</td>
  </tr>
<%
    //利用 while 循环将数据表中的记录列出
  while (rs.next()){
%>
    <tr>
      <td><%=rs.getString("bookid")%></td>
      <td><%=rs.getString("bookname")%></td>
      <td><%=rs.getString("author")%></td>
      <td><%=rs.getString("publisher")%></td>
      <td><%=rs.getString("pubdate")%></td>
      <td><%=rs.getString("price")%></td>
      <td><%=rs.getString("type")%></td>
      <td><%=rs.getString("quantity")%></td>
    </tr>
<%
  }
  rs.close();      //关闭 ResultSet 对象
}
  catch(Exception e){
  out.println(e.getMessage());
  }
  stmt.close();  //关闭 Statement 对象
  conn.close();  //关闭 Connection 对象
%>
</table></center>
</body></html>
```

代码 ex9-03.jsp 在浏览器中的显示效果如图 9-18 所示，顺序输出了 booktable 表中



bookid	bookname	author	publisher	pubdate	price	type	quantity
ISBN 7-04-012301-0	C++程序设计	吴乃陵	高等教育出版社	2003-8-1	29.5	计算机	100
ISBN 7-302-08599-4	C++程序设计	谭浩强	清华大学出版社	2004-6-1	36	计算机	100
ISBN 7-35789-925-256	Google搜索从入门到精通	网友	电子工业出版社	2004-7-27	25	计算机	150
ISBN 7-5053-9856-3	ASP&ASP.NET应用编程150例	王兴东	电子工业出版社	2004-5-1	35	计算机	200
ISBN 7-5606-0939-2	Internet/Intranet应用与网页制作实训教材	樊月华	西安电子科技大学出版社	2003-1-1	15	计算机	100
ISBN 7-5640-0139-9	CPA会计	武玉荣	电子工业出版社	2003-3-1	30	经济	100
ISBN 7-5640-0165-8	现代汽车动力传动装置的控制技术	林学东	清华大学出版社	2004-7-27	30	机械	150
ISBN 7-5640-0244-1	火星的故事	[英]帕特里克.摩尔	人民邮电出版社	2004-5-1	15	小说	100
ISBN 978-7-04-021459-8	计算机应用基础	刘栖丽	高等教育出版社	2007-6-1	29.5	计算机	100
ISBN 978-7-121-05285	计算机网络工程与实训教程	徐武	电子工业出版社	2008-1-1	28	计算机	100
ISBN 978-7-302-12927-4	Web程序设计	(美)Robert W.Sebesta	清华大学出版社	2006-8-1	58	计算机	100
ISBN 978-7-302-15261-3	Web技术	Jeffrey C.Jackson	清华大学出版社	2007-6-1	59	计算机	100
ISBN 978-7-900222-91-6	计算机网络	谢希仁	电子工业出版社	2008-2-1	29	计算机	200

图 9-18　顺序输出了“booktable”表中的所有记录和字段

的所有记录和字段。

9.3.2 参数查询

数据筛选是指按条件从数据库中选出符合条件的所有记录，由 where 子句指定选择记录时要满足的条件。

例 9.4

① 任务要求：在 ex9-04.html 界面输入查询条件(见图 9-19)，如出版社的名称，输入的名称提交给 ex9-04.jsp 处理，输出数据表“booktable”中符合查询条件的图书。其关键查询语句为：



图 9-19 参数查询界面

```
"Select * From booktable where publisher='"+publishername+"'"
```

② ex9-04.html 代码清单如下：

```
<html><head>
     <title>参数查询应用案例</title>
</head>
<body><center>
<font size=4 color=blue>图书查询</font></center><hr>
<form method="post" action="ex9-04.jsp"><font color=green>
  请输入出版社名称：<input type=text name="pubname" size=20  maxlength=20><p>
  <input type=submit value="提交">
  <input type=reset value="清除">
</form></font>
</body></html>
```

③ ex9-04.jsp 代码清单如下：

```
<%@ page contentType="text/html; charset=GB2312"%>
<%@ page import="java.sql.*"%>
<html>
<head><title>选择查询</title>
</head>
<body><center>
<% request.setCharacterEncoding("GB2312");
     String  publishername=request.getParameter("pubname");
     if(publishername==null){
        publishername="";
     }
%>
```

```
<font size=4 color=blue>输出数据表 booktable 中<%=publishername%>的记录</font><hr>
<%
   Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
   Connection conn=DriverManager.getConnection("jdbc:odbc:bookshoplk","sa","");
   Statement stmt=conn.createStatement();
   try{
   ResultSet rs;                        //建立 ResultSet(结果集)对象
   rs=stmt.executeQuery("Select * From booktable where publisher='"+publishername+"'");
   //执行 SQL 语句
%>
<table border=3>
   <tr bgcolor=silver><b>
       <td>bookid</td><td>bookname</td><td>author</td><td>publisher</td>
       <td>pubdate</td><td>price</td><td>&nbsptype</td><td>quantity</td>
   </tr>
<%
       //利用 while 循环将数据表中的记录列出
   while (rs.next()){
%>
      <tr>
       <td> <%=rs.getString("bookid")%></td>
       <td> <%=rs.getString("bookname")%></td>
       <td> <%=rs.getString("author")%></td>
       <td> <%=rs.getString("publisher")%></td>
       <td> <%=rs.getString("pubdate")%></td>
       <td> <%=rs.getString("price")%></td>
       <td> <%=rs.getString("type")%></td>
       <td> <%=rs.getString("quantity")%></td>
      </tr>
<%
   }
   rs.close();                          //关闭 ResultSet 对象
}
   catch(Exception e){
   out.println(e.getMessage());
   }
   stmt.close();                        //关闭 Statement 对象
   conn.close();                        //关闭 Connection 对象
%>
</table></center>
</body></html>
```

④ 代码 ex9-04.jsp 在浏览器中的显示效果如图 9-20 所示。



bookid	bookname	author	publisher	pubdate	price	type	quantity
ISBN 7-302-08599-4	C++程序设计	谭浩强	清华大学出版社	2004-6-1	36	计算机	100
ISBN 7-5640-0165-8	现代汽车动力传动装置的控制技术	林学东	清华大学出版社	2004-7-27	30	机械	150
ISBN 978-7-302-12927-4	Web程序设计	(美)Robert W.Sebesta	清华大学出版社	2006-8-1	58	计算机	100
ISBN 978-7-302-15261-3	Web技术	Jeffrey C.Jackson	清华大学出版社	2007-6-1	59	计算机	100

图 9-20　查询某出版社出版的图书

9.3.3　模糊查询

使用 like 语句和通配符进行模糊查询。在模糊查询中使用通配符“%”代表任意多个字符，“_”代表任意一个字符。

例 9.5

① 任务要求：在界面(ex9-05.html)中输入书名中的部分词汇，输入的内容提交给 ex9-05.jsp 处理，输出数据表“booktable”中所有包含该词汇图书的书名。其关键查询语句为：

```
"Select * From booktable where bookname like' %"+b_name+"%'"
```

表示查询书名中包含有 b_name 中内容的图书。

② 代码 ex9-05.html 清单如下：

```
<html><head>
    <title>模糊查询应用案例</title>
</head>
<body><center>
<font size=4 color=blue>模糊查询</font></center><hr>
<form method="post" action="ex9-05.jsp"><font color=green>
  书名：<input type=text name="bookname" size=20  maxlength=20><br>
  注：可以输入部分词汇<br>
  <input type=submit value="提交">
  <input type=reset value="清除">
</form></font>
</body></html>
```

③ 代码 ex9-05.jsp 清单如下：

```
<%@ page contentType="text/html; charset=GB2312"%>
<%@ page import="java.sql.*"%>
<html>
<head><title>模糊查询</title>
</head>
```

```
<body><center>
<%request.setCharacterEncoding("GB2312");
   String  b_name=request.getParameter("bookname");
   if(b_name==null){
     b_name="";
   }
%>
<font size=4 color=blue>输出与<%=b_name%>有关的图书</font><hr>
<%
  Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
  Connection conn=DriverManager.getConnection("jdbc:odbc:bookshoplk","sa","");
  Statement stmt=conn.createStatement();
  try{
  ResultSet rs; //建立 ResultSet(结果集)对象
  rs=stmt.executeQuery("Select * From booktable where bookname like '%"+b_name+"%'");
  //执行 SQL 语句
%>
<table border=3>
  <tr bgcolor=silver><b>
    <td>bookid</td><td>bookname</td><td>author</td><td>publisher</td>
    <td>pubdate</td><td>price</td><td>&nbsptype</td><td>quantity</td>
  </tr>
<%
    //利用 while 循环将数据表中的记录列出
  while (rs.next()){
%>
  <tr>
    <td><%=rs.getString("bookid") %></td>
    <td><%=rs.getString("bookname")%></td>
    <td><%=rs.getString("author")%></td>
    <td><%=rs.getString("publisher")%></td>
    <td><%=rs.getString("pubdate")%></td>
    <td><%=rs.getString("price")%></td>
    <td><%=rs.getString("type")%></td>
    <td><%=rs.getString("quantity")%></td>
  </tr>
<%
  }
  rs.close();   //关闭 ResultSet 对象
}
  catch(Exception e){
  out.println(e.getMessage());
  }
  stmt.close();    //关闭 Statement 对象
  conn.close();    //关闭 Connection 对象
%>
```

```
</table></center>
</body></html>
```

④ 在浏览器中的显示效果如图 9-21 所示。



bookid	bookname	author	publisher	pubdate	price	type	quantity
ISBN 7-04-012301-0	C++程序设计	吴乃陵	高等教育出版社	2003-8-1	29.5	计算机	100
ISBN 7-111-08318-0	C++程序设计教程	郑莉	机械工业出版社	2003-2-1	28	计算机	100
ISBN 7-302-08599-4	C++程序设计	谭浩强	清华大学出版社	2004-6-1	36	计算机	100

图 9-21 模糊查询

9.3.4 范围查询

范围查询是查找值在数据表中某一范围内的数据。

例 9.6

① 任务要求：输出数据表 booktable 中某段时间内出版的图书。在界面(ex9-06.html，见图 9-22)中输入查询的开始日期和截止日期，提交给 ex9-06.jsp 处理，输出数据表 booktable 中该时间段出版的图书。其关键查询语句为：

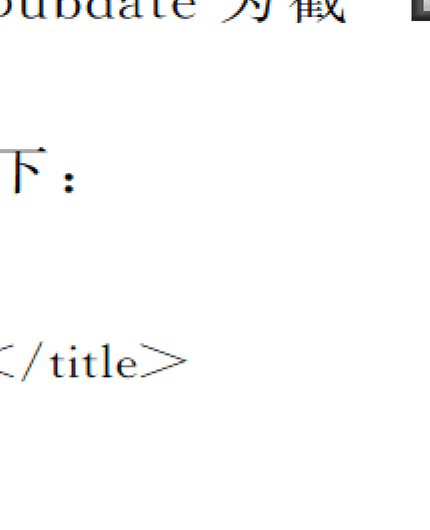
"Select * From booktable where pubdate between '"+s_pubdate+"' and '"+e_pubdate+"'"

其中 s_pubdate 为起始日期，e_pubdate 为截止日期。



图 9-22 范围查询

② 代码 ex9-06.html 清单如下：



```
<html><head>
    <title>范围查询应用案例</title>
</head>
<body><center>
<font size=4 color=blue>根据出版日期查询图书</font></center><hr>
<form method="post" action="ex9-06.jsp"><font color=green>
  查询在<input type=text name="starpubdate" size=10  maxlength=10>
  和<input type=text name="endpubdate" size=10  maxlength=10>之间出版的图书<p>
  <input type=submit value="提交">
  <input type=reset value="清除">
</form></font>
</body></html>
```

③ 代码 ex9-06.jsp 清单如下：

```
<%@ page contentType="text/html; charset=GB2312"%>
<%@ page import="java.sql.*"%>
<html>
<head><title>范围查询</title>
</head>
<body><center>
<%request.setCharacterEncoding("GB2312");
    String  s_pubdate=request.getParameter("starpubdate");
    if(s_pubdate==null){
      s_pubdate="";
    }
    String  e_pubdate=request.getParameter("endpubdate");
    if(e_pubdate==null){
      e_pubdate="";
    }
%>
<font size=4 color=blue>在<%=s_pubdate%>和<%=e_pubdate%>之间出版的图书
</font><hr>
<%
  Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
  Connection conn=DriverManager.getConnection("jdbc:odbc:bookshoplk","sa","");
  Statement stmt=conn.createStatement();
  try{
  ResultSet rs; //建立 ResultSet(结果集)对象
  rs=stmt.executeQuery("Select * From booktable where pubdate
                               between '"+s_pubdate+"' and '"+e_pubdate+"'");
  //执行 SQL 语句
%>
<table border=3>
  <tr bgcolor=silver><b>
    <td>bookid</td><td>bookname</td><td>author</td><td>publisher</td>
    <td>pubdate</td><td>price</td><td>&nbsptype</td><td>quantity</td>
  </tr>
<%
    //利用 while 循环将数据表中的记录列出
  while (rs.next()){
%>
  <tr>
    <td><%=rs.getString("bookid")%></td>
    <td><%=rs.getString("bookname")%></td>
    <td><%=rs.getString("author")%></td>
    <td><%=rs.getString("publisher")%></td>
```

```
        <td><%=rs.getString("pubdate")%></td>
        <td><%=rs.getString("price")%></td>
        <td><%=rs.getString("type")%></td>
        <td><%=rs.getString("quantity")%></td>
    </tr>
<%
    }
    rs.close();         //关闭 ResultSet 对象
}
    catch(Exception e){
    out.println(e.getMessage());
    }
    stmt.close();     //关闭 Statement 对象
    conn.close();     //关闭 Connection 对象
%>
</table></center>
</body></html>
```

④ 代码 ex9-06.jsp 在浏览器中的显示效果如图 9-23 所示。

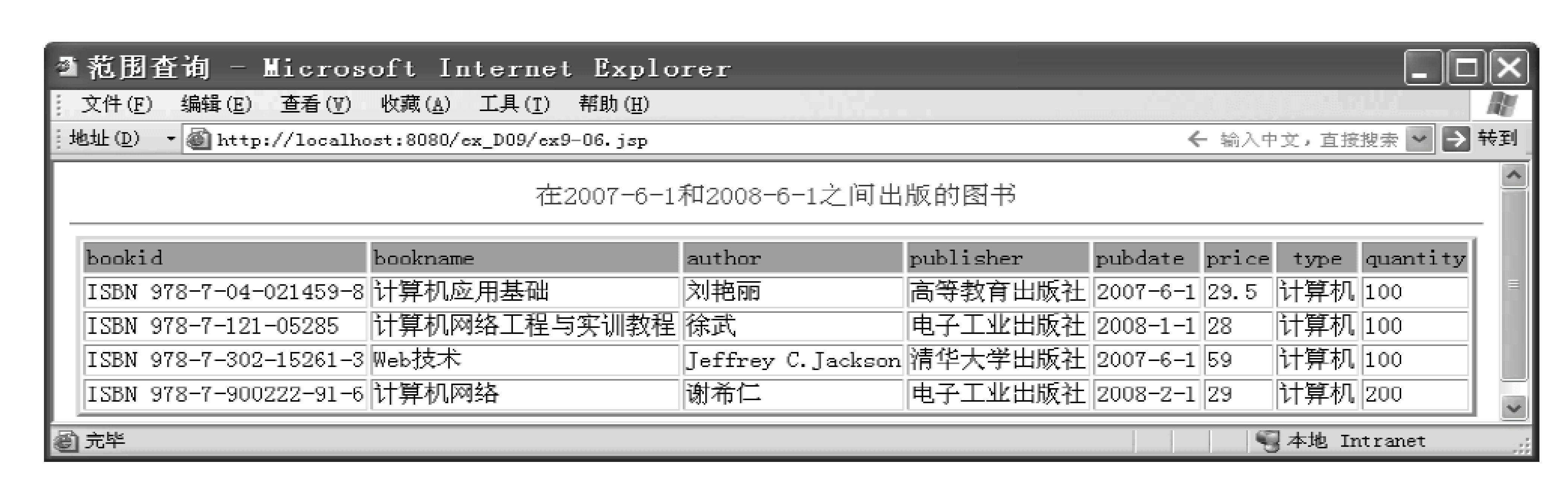


图 9-23　输出在 2007-6-1 到 2008-6-1 之间出版的图书

9.3.5　复合条件查询

复合条件查询是多个查询条件的查询。

例 9.7

① 任务要求：输出数据表 booktable 中某个类别、某时间后出版的图书。在界面(ex9-07.html，见图 9-24)中输入类别和日期，提交给 ex9-07.jsp 处理，输出符合查询条件的图书。其关键查询语句为：

″Select * From booktable where type=′″+b_typename+″′ and pubdate>=′″+e_pubdate+″′″



图 9-24　复合条件查询

② 代码 ex9-07.html 清单如下：

```
<html><head>
    <title>复合查询应用案例</title>
</head>
<body><center>
<font size=4 color=blue>根据类别和出版日期查询图书</font></center><hr>
<form method="post" action="ex9-07.jsp"><font color=green>
  查询<input type=text name="typename" size=10  maxlength=10>类
  <input type=text name="endpubdate" size=10  maxlength=10>之后出版的图书<p>
  <input type=submit value="提交">
  <input type=reset value="清除">
</form></font>
</body></html>
```

③ 代码 ex9-07.jsp 清单如下：

```
<%@ page contentType="text/html; charset=GB2312"%>
<%@ page import="java.sql.*"%>
<html>
<head><title>复合查询</title>
</head>
<body><center>
<%request.setCharacterEncoding("GB2312");
    String  b_typename=request.getParameter("typename");
    if(b_typename==null){
      b_typename="";
    }
    String  e_pubdate=request.getParameter("endpubdate");
    if(e_pubdate==null){
      e_pubdate="";
    }
%>
<font size=4 color=blue>查询<%=b_typename%>类在<%=e_pubdate%>后出版的图书
</font><hr>
<%
  Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
  Connection conn=DriverManager.getConnection("jdbc:odbc:bookshoplk","sa","");
  Statement stmt=conn.createStatement();
  try{
  ResultSet rs; //建立 ResultSet(结果集)对象
  rs=stmt.executeQuery("Select * From booktable where
                          type='"+b_typename+"' and pubdate>='"+e_pubdate+"'");
  //执行 SQL 语句
%>
```

```
<table border=3>
  <tr bgcolor=silver><b>
    <td>bookid</td><td>bookname</td><td>author</td><td>publisher</td>
    <td>pubdate</td><td>price</td><td>&nbsptype</td><td>quantity</td>
  </tr>
<%
    //利用 while 循环将数据表中的记录列出
  while (rs.next()){
%>
  <tr>
    <td><%=rs.getString("bookid")%></td>
    <td><%=rs.getString("bookname")%></td>
    <td><%=rs.getString("author")%></td>
    <td><%=rs.getString("publisher")%></td>
    <td><%=rs.getString("pubdate")%></td>
    <td><%=rs.getString("price")%></td>
    <td><%=rs.getString("type")%></td>
    <td><%=rs.getString("quantity")%></td>
  </tr>
<%
  }
  rs.close();      //关闭 ResultSet 对象
}
  catch(Exception e){
  out.println(e.getMessage());
  }
  stmt.close();  //关闭 Statement 对象
  conn.close();  //关闭 Connection 对象
%>
</table></center>
</body></html>
```

④ 代码 ex9-07.jsp 在浏览器中的显示效果如图 9-25 所示。



bookid	bookname	author	publisher	pubdate	price	type	quantity
ISBN 978-7-121-05285	计算机网络工程与实训教程	徐武	电子工业出版社	2008-1-1	28	计算机	100
ISBN 978-7-900222-91-6	计算机网络	谢希仁	电子工业出版社	2008-2-1	29	计算机	200

图 9-25 输出 2008-1-1 以后出版的计算机类图书

9.3.6 排序查询

在 SQL 语句中应用 Order By 子语句，对查询结果记录进行排序。

例 9.8

① 任务要求：在界面(ex9-08.html，见图 9-26)选择排序项目，提交给 ex9-08.jsp 处理，根据用户要求排序输出图书。其关键查询语句为：

```
"Select * From booktable Order By "+s_rname+""
```



图 9-26 选择排序方式

表示取出 booktable 表中的所有数据，并按 s_rname 所指明的字段排序。s_rname 是用户选择的排序字段。

② 代码 ex9-08.html 清单如下：

```
<html><head>
    <title>排序查询应用案例</title>
</head>
<body><center>
<font size=4 color=blue>排序查询</font></center><hr>
<form method="post" action="ex9-08.jsp"><font color=green>
  <input type=radio name="rname" value="pubdate" checked>按出版日期排序
  <input type=radio name="rname" value="bookid">按书号排序
  <input type=radio name="rname" value="price">按价格排序<p>
  <input type=submit value="提交">
  <input type=reset value="清除">
</form></font>
</body></html>
```

③ 代码 ex9-08.jsp 清单如下：

```
<%@ page contentType="text/html; charset=GB2312"%>
<%@ page import="java.sql.*"%>
<html>
```

```
<head><title>排序查询</title>
</head>
<body><center>
<%request.setCharacterEncoding("GB2312");
    String  s_rname=request.getParameter("rname");
    if(s_rname==null){
      s_rname="";
    }
%>
<font size=4 color=blue>按<%=s_rname%>排序</font><hr>
<%
  Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
  Connection conn=DriverManager.getConnection("jdbc:odbc:bookshoplk","sa","");
  Statement stmt=conn.createStatement();
  try{
  ResultSet rs;          //建立 ResultSet(结果集)对象
  rs=stmt.executeQuery("Select * From booktable Order By "+s_rname+"Desc");
  //执行 SQL 语句
%>
<table border=3>
  <tr bgcolor=silver><b>
    <td>bookid</td><td>bookname</td><td>author</td><td>publisher</td>
    <td>pubdate</td><td>price</td><td>&nbsptype</td><td>quantity</td>
  </tr>
<%
    //利用 while 循环将数据表中的记录列出
  while (rs.next()){
%>
  <tr>
    <td><%=rs.getString("bookid")%></td>
    <td><%=rs.getString("bookname")%></td>
    <td><%=rs.getString("author")%></td>
    <td><%=rs.getString("publisher")%></td>
    <td><%=rs.getString("pubdate")%></td>
    <td><%=rs.getString("price")%></td>
    <td><%=rs.getString("type")%></td>
    <td><%=rs.getString("quantity")%></td>
  </tr>
<%
  }
  rs.close();            //关闭 ResultSet 对象
}
  catch(Exception e){
```

```
        out.println(e.getMessage());
    }
    stmt.close();          //关闭 Statement 对象
    conn.close();          //关闭 Connection 对象
%>
</table></center>
</body></html>
```

④ 代码 ex9-08.jsp 在浏览器中的显示效果如图 9-27 所示，图书按用户要求的“出版日期”降序排序。

排序查询 - Microsoft Internet Explorer

文件(F) 编辑(E) 查看(V) 收藏(A) 工具(T) 帮助(H)

地址(D) http://localhost:8080/ex_D09/ex9-08.jsp

按pubdate排序

bookid	bookname	author	publisher	pubdate	price	type	quantity
ISBN 978-7-900222-91-6	计算机网络	谢希仁	电子工业出版社	2008-2-1	29	计算机	200
ISBN 978-7-121-05285	计算机网络工程与实训教程	徐武	电子工业出版社	2008-1-1	28	计算机	100
ISBN 978-7-04-021459-8	计算机应用基础	刘艳丽	高等教育出版社	2007-6-1	29.5	计算机	100
ISBN 978-7-302-15261-3	Web技术	Jeffrey C.Jackson	清华大学出版社	2007-6-1	59	计算机	100
ISBN 978-7-302-12927-4	Web程序设计	(美)Robert W.Sebesta	清华大学出版社	2006-8-1	58	计算机	100
ISBN 7-35789-925-256	Google搜索从入门到精通	网友	电子工业出版社	2004-7-27	25	计算机	150
ISBN 7-5640-0165-8	现代汽车动力传动装置的控制技术	林学东	清华大学出版社	2004-7-27	30	机械	150
ISBN 7-302-08599-4	C++程序设计	谭浩强	清华大学出版社	2004-6-1	36	计算机	100
ISBN 7-5053-9856-3	ASP&ASP.NET应用编程150例	王兴东	电子工业出版社	2004-5-1	35	计算机	200
ISBN 7-5640-0244-1	火星的故事	[英]帕特里克.摩尔	人民邮电出版社	2004-5-1	15	小说	100
ISBN 7-04-012301-0	C++程序设计	吴乃陵	高等教育出版社	2003-8-1	29.5	计算机	100
ISBN 7-5640-0139-9	CPA会计	武玉荣	电子工业出版社	2003-3-1	30	经济	100
ISBN 7-111-08318-0	C++程序设计教程	郑莉	机械工业出版社	2003-2-1	28	计算机	100
ISBN 7-5606-0939-2	Internet/Intranet应用与网页制作实训教材	樊月华	西安电子科技大学出版社	2003-1-1	15	计算机	100

完毕　本地 Intranet

图 9-27　按出版日期输出图书记录

9.4　添加记录

1. 使用 SQL 语句添加新记录

使用 SQL 语句在数据库表中添加新记录。

例 9.9　管理员在界面(ex9-09.html，见图 9-28)输入需要添加到数据库新书的数据，并把这些数据提交给 ex9-09.jsp 处理。在 ex9-09.jsp 中用 SQL 的 Insert 命令向 booktable 数据表插入一条新的图书记录，并显示新添加的记录。其关键语句为：

```
Insert Into booktable(bookid,bookname,author,publisher,pubdate,price,type,quantity)
Values('"+s_bkid+"','"+s_bkname+"','"+s_authorname+"','"+s_bkpublisher+"','
        "+s_bkpubdate+"','"+s_bkprice+"','"+s_bktype+"',"+s_bkquantity+")";
```

该语句将在 booktable 表中插入一条记录，其中 bookid 字段的值为 s_bkid 项的值，bookname 的值为 s_bkname 项的值，其余类推。

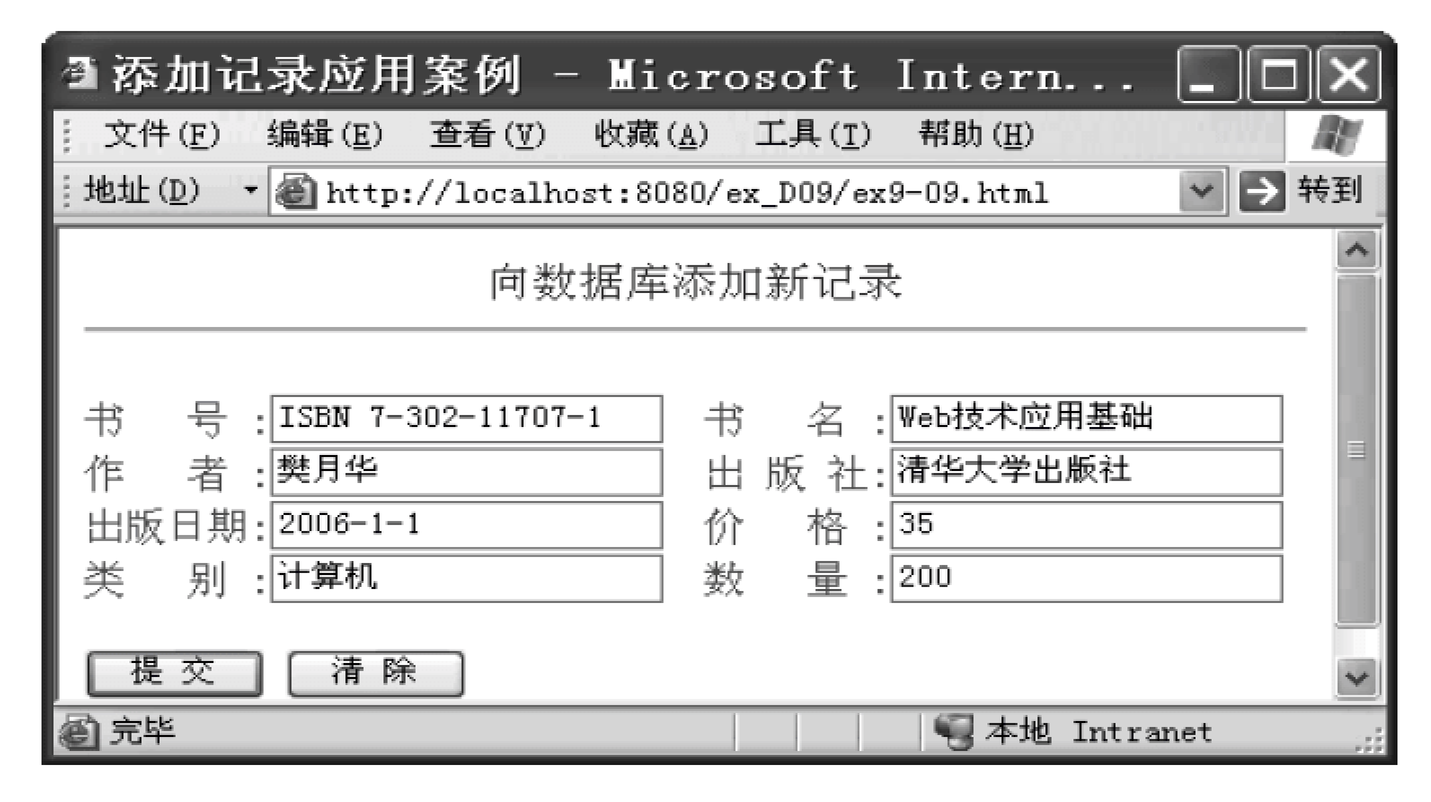


图 9-28　添加新记录

2. ex9-09.html 代码清单

```
<html><head>
     <title>添加记录应用案例</title>
</head>
<body><center>
<font size=4 color=blue>向数据库添加新记录</font></center><hr>
<form method="post" action="ex9-09.jsp"><font color=green>
   书号：<input type=text name="bkid" size=20>
   书名：<input type=text name="bkname" size=20><br>
   作者：<input type=text name="authorname" size=20>
   出版社：<input type=text name="bkpublisher" size=20><br>
   出版日期：<input type=text name="bkpubdate" size=20>
   价格：<input type=text name="bkprice" size=20><br>
   类别：<input type=text name="bktype" size=20>
   数量：<input type=text name="bkquantity" size=20><p>
   <input type=submit value="提交">
   <input type=reset value="清除">
</form></font>
</body></html>
```

3. ex9-09.jsp 代码清单

```
<%@ page contentType="text/html; charset=GB2312"%>
<%@ page import="java.sql.*"%>
<html>
<head><title>添加记录</title>
</head>
<body><center>
<%request.setCharacterEncoding("GB2312");
```

```
    String  s_bkid=request.getParameter("bkid");
    if(s_bkid==null){
      s_bkid="";
    }
    String  s_bkname=request.getParameter("bkname");
    if(s_bkname==null){
      s_bkname="";
    }
    String  s_authorname=request.getParameter("authorname");
    if(s_authorname==null){
      s_authorname="";
    }
    String  s_bkpublisher=request.getParameter("bkpublisher");
    if(s_bkpublisher==null){
      s_bkpublisher="";
    }
    String  s_bkpubdate=request.getParameter("bkpubdate");
    if(s_bkpubdate==null){
      s_bkpubdate="";
    }
    String  s_bkprice=request.getParameter("bkprice");
    if(s_bkprice==null){
      s_bkprice="";
    }
    String  s_bktype=request.getParameter("bktype");
    if(s_bktype==null){
      s_bktype="";
    }
    String  s_bkquantity=request.getParameter("bkquantity");
    if(s_bkquantity==null){
      s_bkquantity="";
    }
%>
<font size=4 color=blue>新添加的记录</font><hr>
<%
  String sql;
  Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
  Connection conn=DriverManager.getConnection("jdbc:odbc:bookshoplk","sa","");
  Statement stmt=conn.createStatement();
  try{
  sql="Insert Into booktable(bookid,bookname,author,publisher,pubdate,price,type,quantity)
       Values('"+s_bkid+"','"+s_bkname+"','"+s_authorname+"','"+s_bkpublisher
       +"','"+s_bkpubdate+"','"+s_bkprice+"','"+s_bktype+"',"+s_bkquantity+")";
```

```
    stmt.executeUpdate(sql);
    ResultSet rs;        //建立 ResultSet(结果集)对象
    rs=stmt.executeQuery("Select * From booktable where bookid='"+s_bkid+"'");
    //执行 SQL 语句
%>
<table border=3>
    <tr bgcolor=silver><b>
      <td>bookid</td><td>bookname</td><td>author</td><td>publisher</td>
      <td>pubdate</td><td>price</td><td>type</td><td>quantity</td>
    </tr>
<%
      //利用 while 循环将数据表中的记录列出
    while(rs.next()){
%>
    <tr>
      <td><%=rs.getString("bookid")%></td>
      <td><%=rs.getString("bookname")%></td>
      <td><%=rs.getString("author")%></td>
      <td><%=rs.getString("publisher")%></td>
      <td><%=rs.getString("pubdate")%></td>
      <td><%=rs.getString("price")%></td>
      <td><%=rs.getString("type")%></td>
      <td><%=rs.getString("quantity")%></td>
    </tr>
<%
    }
    rs.close();          //关闭 ResultSet 对象
}
    catch(Exception e){
    out.println(e.getMessage());
    }
    stmt.close();        //关闭 Statement 对象
    conn.close();        //关闭 Connection 对象
%>
</table></center>
</body></html>
```

4. 运行结果

代码 ex9-09.jsp 在浏览器中的显示效果如图 9-29 所示。

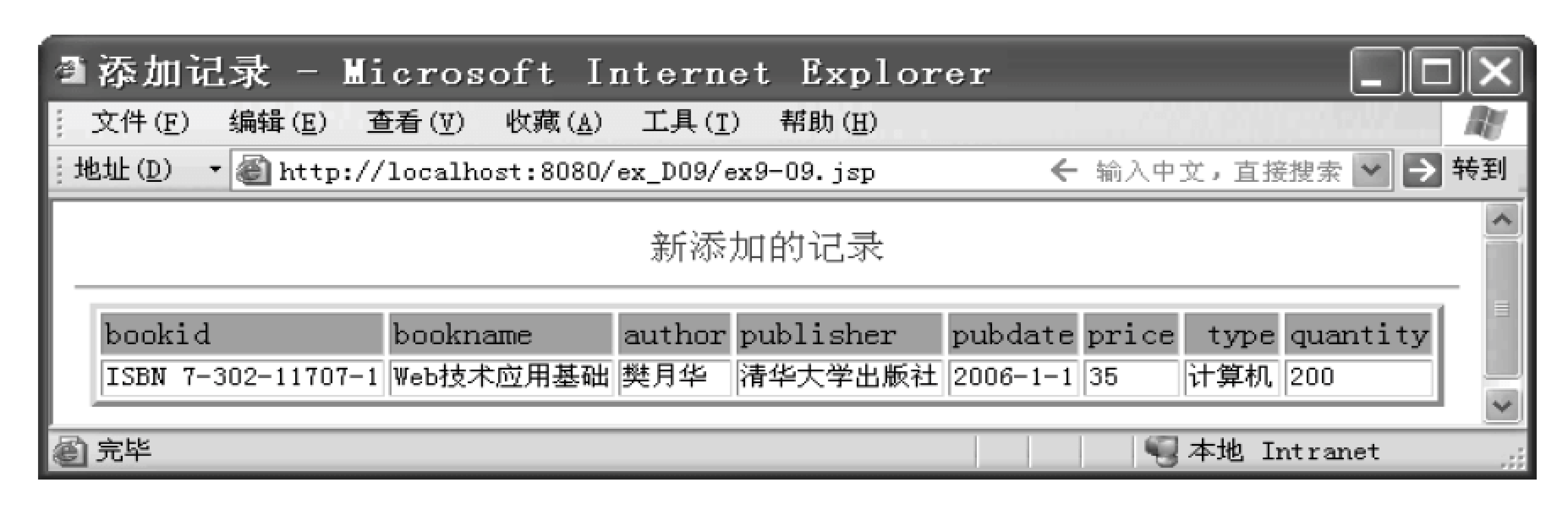


图 9-29　添加并显示新记录

9.5　更新记录

1. 使用 SQL 语句更新记录

使用 SQL 语句更新数据库中的记录，通过 where 子句限定要进行更新的记录。

例 9.10　管理员在界面(ex9-10.html，见图 9-30)输入需要更新数据的图书书号，并把更新的数据提交给 ex9-10.jsp 处理。在 ex9-10.jsp 中用 SQL 的 Update 语句更新记录，并输出更新后的记录。其关键语句为：

```
"update booktable Set quantity='"+s_bkquantity+"' where bookid='"+s_bkid+"'";
```



图 9-30　更新记录

表示更新 booktable 中的记录，条件是该书的 bookid 等于用户输入的图书号，该书的数量等于用户输入的数量。

2. ex9-10.html 代码清单

```
<html><head>
    <title>更新记录应用案例</title>
</head>
<body><center>
<font size=4 color=blue>更新记录</font></center><hr>
```

```
<form method="post" action="ex9-10.jsp"><font color=green>
   输入需要更新数量的图书书号：<input type=text name="bkid" size=20> 
   输入新的数量：<input type=text name="bkquantity" size=20><p>
   <input type=submit value="提交">
   <input type=reset value="清除">
</form></font>
</body></html>
```

3. ex9-10.jsp 代码清单

```
<%@ page contentType="text/html; charset=GB2312"%>
<%@ page import="java.sql.*"%>
<html>
<head><title>更新记录</title>
</head>
<body><center>
<%request.setCharacterEncoding("GB2312");
    String  s_bkid=request.getParameter("bkid");
    if(s_bkid==null){
      s_bkid="";
    }
    String  s_bkquantity=request.getParameter("bkquantity");
    if(s_bkquantity==null){
      s_bkquantity="";
    }
%>
<font size=4 color=blue>更新记录</font><hr>
<%
   String sql;
   Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
   Connection conn=DriverManager.getConnection("jdbc:odbc:bookshoplk","sa","");
   Statement stmt=conn.createStatement();
   try{
   sql="update booktable Set quantity='"+s_bkquantity+"' where bookid='"+s_bkid+"'";
   stmt.executeUpdate(sql);
   ResultSet rs; //建立 ResultSet(结果集)对象
   rs=stmt.executeQuery("Select * From booktable where bookid='"+s_bkid+"'");
   //执行 SQL 语句
%>
<table border=3>
   <tr bgcolor=silver><b>
     <td>bookid</td><td>bookname</td><td>author</td><td>publisher</td>
```

```
    <td>pubdate</td><td>price</td><td>type</td><td>quantity</td>
  </tr>
<%
    //利用 while 循环将数据表中的记录列出
  while (rs.next()){
%>
  <tr>
    <td><%=rs.getString("bookid")%></td>
    <td><%=rs.getString("bookname")%></td>
    <td><%=rs.getString("author")%></td>
    <td><%=rs.getString("publisher")%></td>
    <td><%=rs.getString("pubdate")%></td>
    <td><%=rs.getString("price")%></td>
    <td><%=rs.getString("type")%></td>
    <td><%=rs.getString("quantity")%></td>
  </tr>
<%
  }
  rs.close();      //关闭 ResultSet 对象
}
  catch(Exception e){
  out.println(e.getMessage());
  }
  stmt.close();  //关闭 Statement 对象
  conn.close();  //关闭 Connection 对象
%>
</table></center>
</body></html>
```

4. 更新结果

代码 ex9-10.jsp 在浏览器中的显示效果如图 9-31 所示，书号为 ISBN 7-302-11707-1 的图书已经更新为 200 册。

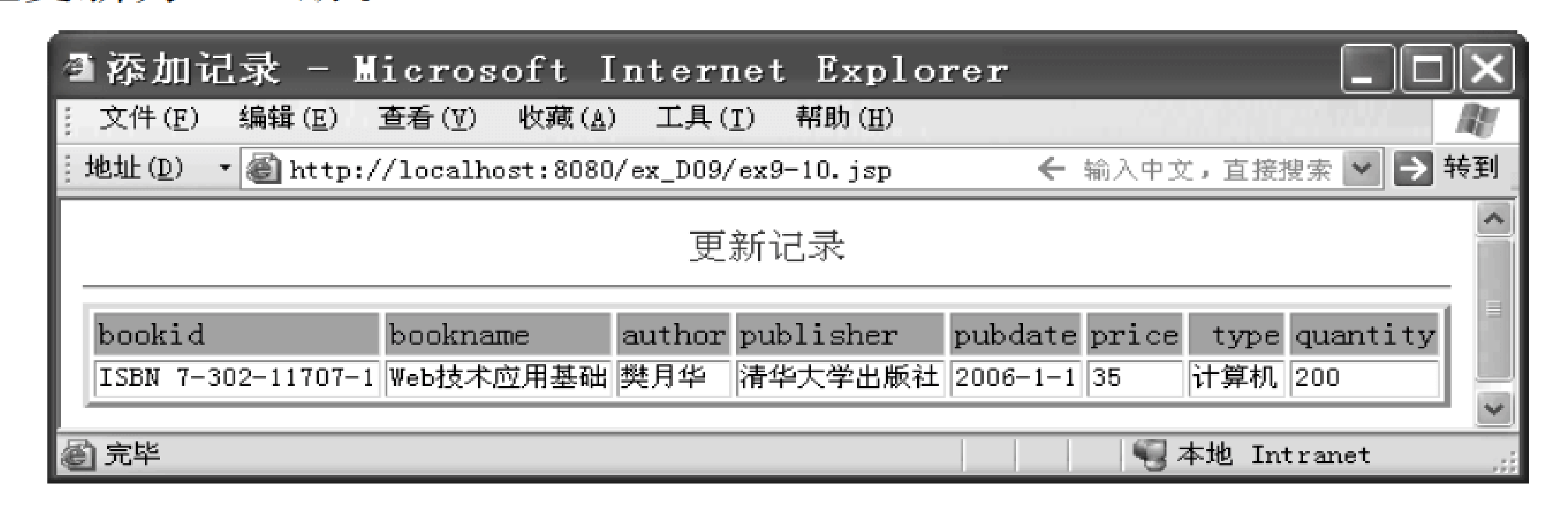


图 9-31 更新后的结果

9.6 删除记录

1. 使用SQL语句删除记录

例9.11 管理员在界面(ex9-11.html,见图9-32)输入需要删除图书的书号,并把要删除的书号数据提交给ex9-11.jsp处理。在ex9-11.jsp中,用SQL的Delete语句将该记录删除,并显示删除后的booktable数据表。其关键语句为:

```
"Delete From booktable Where bookid='"+s_bkid+"'";
```



图9-32 删除记录

表示删除booktable表中的bookid为用户输入书号(s_bkid)的图书。

本例删除刚才添加的书号为ISBN 7-302-11707-1的图书。

2. ex9-11.html代码清单

```
<html><head>
    <title>删除记录应用案例</title>
</head>
<body><center>
<font size=4 color=blue>删除记录</font></center><hr>
<form method="post" action="ex9-11.jsp"><font color=green>
  输入需要删除的图书书号:<input type=text name="bkid" size=20><p>
  <input type=submit value="提交">
  <input type=reset value="清除">
</form></font>
</body></html>
```

3. ex9-11.jsp代码清单

```
<%@ page contentType="text/html; charset=GB2312"%>
```

```
<%@ page import="java.sql.*"%>
<html>
<head><title>删除记录</title>
</head>
<body><center>
<%request.setCharacterEncoding("GB2312");
    String  s_bkid=request.getParameter("bkid");
    if(s_bkid==null){
      s_bkid="";
    }
%>
<font size=4 color=blue>删除后的记录</font><hr>
<%
   String sql;
   Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
   Connection conn=DriverManager.getConnection("jdbc:odbc:bookshoplk","sa","");
   Statement stmt=conn.createStatement();
   try{
   sql="Delete From booktable Where bookid='"+s_bkid+"'";
   stmt.executeUpdate(sql);
   ResultSet rs; //建立 ResultSet(结果集)对象
   rs=stmt.executeQuery("Select * From booktable");
   //执行 SQL 语句
%>
<table border=3>
   <tr bgcolor=silver><b>
      <td>bookid</td><td>bookname</td><td>author</td><td>publisher</td>
      <td>pubdate</td><td>price</td><td>type</td><td>quantity</td>
   </tr>
<%
      //利用 while 循环将数据表中的记录列出
   while (rs.next()){
%>
   <tr>
      <td><%=rs.getString("bookid")%></td>
      <td><%=rs.getString("bookname")%></td>
      <td><%=rs.getString("author")%></td>
      <td><%=rs.getString("publisher")%></td>
      <td><%=rs.getString("pubdate")%></td>
      <td><%=rs.getString("price") %></td>
      <td><%=rs.getString("type") %></td>
      <td><%=rs.getString("quantity")%></td>
   </tr>
<%
```

```
    }
    rs.close();        //关闭 ResultSet 对象
}
    catch(Exception e){
    out.println(e.getMessage());
    }
    stmt.close();    //关闭 Statement 对象
    conn.close();    //关闭 Connection 对象
%>
</table></center>
</body></html>
```

4. 删除后的 booktable 数据表

由于删除的是在例 9.9 中添加的记录，所以删除该记录后得到的数据表仍然如图 9-18 所示。

习题、上机练习与实训 9

一、习题

1. 名词解释：数据库、数据库管理系统、管理信息系统。
2. SQL 语句的主要功能是什么？
3. JDBC 是一种什么技术？它的特点和功能是什么？
4. 列举 JDBC 访问数据库 4 种方式中的两种，其中 JDBC-ODBC 桥适用于什么条件？
5. 简要列出 JDBC 建立数据库连接步骤。
6. 请写出包含 java.sql 的 Page 指令语句。
7. 请写出加载 JDBC-ODBC 桥驱动程序的语句。
8. 如果数据库 bookshop 的数据源名是：bookshopdsn，请写出与该数据库建立连接的语句。
9. 用 Statement 类的对象向上题的 bookshop 数据库发送 SQL 语句。

二、上机练习

1. 设计一个网上商店应用(商品种类可以自选)，并为该应用设计一个数据库，至少要有三张表。创建该数据库。
2. 输出网上商店数据库中某张表的表头。
3. 输出网上商店数据库中某张表的第 2 条记录。
4. 输出网上商店数据库中某张表的所有记录。
5. 输出网上商店数据库中某张表中的指定记录。
6. 对网上商店数据库中的记录进行模糊查询。
7. 对网上商店数据库中的记录进行范围查询。

8．把网上商店数据库中某张表的数据逆序输出。

9．对网上商店数据库中某张表进行插入、删除和更新操作。

三、实训课题

1．使用 JSP 技术建立一个班级网站，要求：

（1）建立一个班级数据库，数据库中应有班级的基本信息，如：通信录、成绩单等。

（2）制作一个动态网页。该网页能够接收用户信息，并根据用户要求把后台数据库信息发布到前端的浏览器。

（3）在网页中提供各种查询功能，例如：查询某位同学的学习成绩、计算班级的及格率或优秀率等。

2．为校园网完成一个学籍管理软件中的学生成绩管理模块的设计与制作，包括学生成绩的浏览、录入、添加、删除、查询和更新等功能。

3．在本章上机练习的基础上完成网上商店的应用开发，要求有商品浏览和购物车功能。

第10章　网上书店的实现

网上书店的系统分析与设计已在第3章完成，部分环节的实现也在其他章节进行了介绍，本章将介绍该系统主要功能的实现。网上书店由客户端业务处理和管理端业务处理两个部分组成，其功能结构见第3章的图3-3。

客户端业务处理主要包括用户身份验证、图书展示、购书车等模块。管理端业务处理主要包括图书管理、用户管理、订单管理、留言管理、出版社管理和职工管理等模块。

网上书店的安装见第3章的3.6节。

10.1　主界面实现

10.1.1　客户端处理主界面

1. 页面布局

客户端业务处理主界面见第3章的图3-5。它的页面布局如图10-1所示。

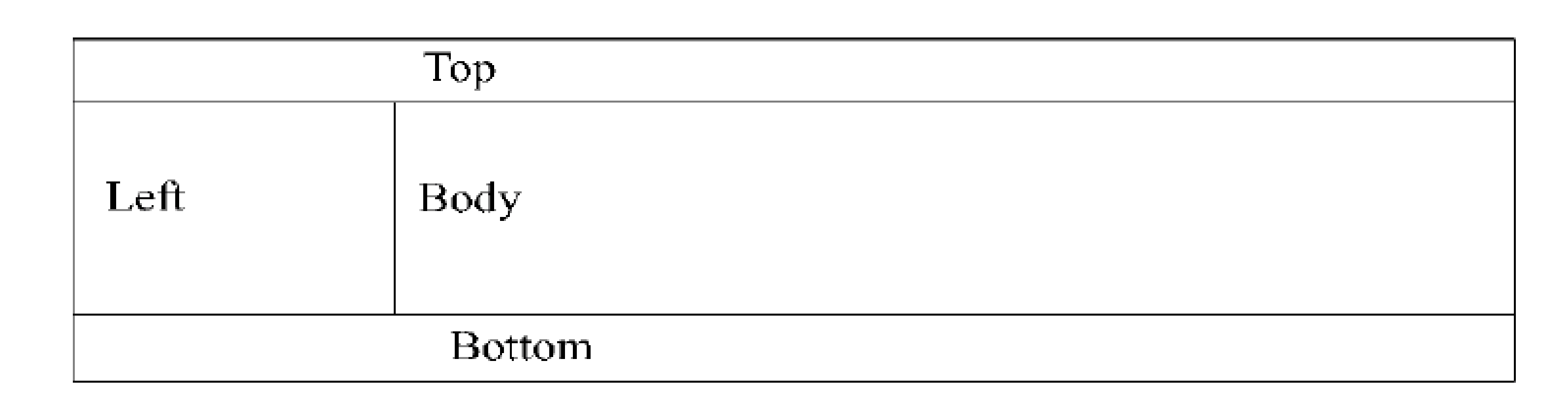


图10-1　客户端业务处理主界面的页面布局

各部分的功能与相关代码如下。

(1) Top

客户端业务处理的Top部分的文件名是top.jsp，使用＜％@include file＝″top.jsp″％＞语句把它嵌入客户端业务处理各个页面的顶部，使得系统具有统一的风格。Top部分用来显示logo图片、日期和客户端各功能模块的入口。各功能模块和它们对应的程序如下：

- 首页——index.jsp

- 精品图书——excellent.jsp
- 新书架——newbook.jsp
- 书目查找——booksearch.jsp
- 我的订单——myorder.jsp
- 购书车——shoppingcart.jsp
- 读者留言——leaveword.jsp

(2) Left

Left 用来显示公告栏、用户登录和图书检索等信息。各功能模块和它们对应的程序如下：

- 公告栏——declare.jsp
- 用户登录——login.jsp
- 图书检索——search.jsp

(3) Body

Body 为当前页面的主要内容显示区域。

(4) Bottom

Bottom 部分的文件名是 bottom.jsp，使用<%@include file="bottom.jsp"%>语句嵌入网上书店的各个页面的底部，使得系统具有统一的风格。Bottom 部分主要用来显示版权信息、联系信息等。

2. 主界面 index.jsp 代码清单

主界面代码清单见 bookshop 目录下的 index.jsp 文件，它的框架结构代码清单见第 4 章的 4.9 节。

10.1.2 管理端处理主界面

管理端业务处理主界面见第 3 章的图 3-7。它的页面布局见图 10-2。

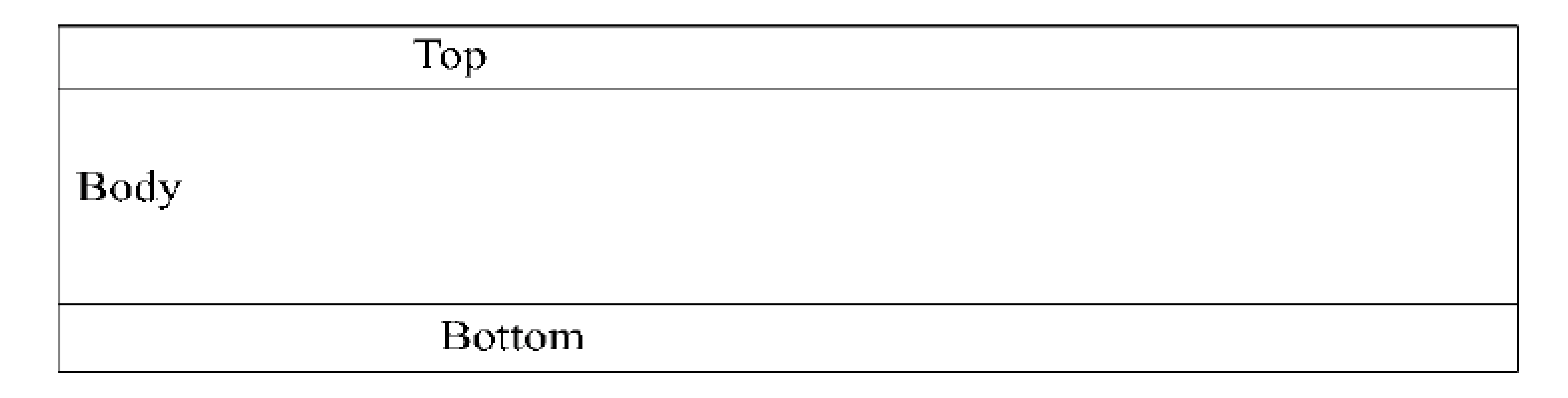


图 10-2 管理端业务处理主界面的页面布局

各部分的功能与相关代码如下。

(1) Top

管理端业务处理主界面 Top 部分的文件名也是 top.jsp，但是与客户端业务处理的 top.jsp 文件在不同的目录下。应用<%@include file="top.jsp"%>语句，把它嵌入管理端业务处理主界面各个页面的顶部，使得系统具有统一的风格。Top 部分用来显示

logo 图片、日期和管理端各功能模块的入口。各功能模块和它们对应的程序如下：

- 图书管理——booklist.jsp
- 用户管理——userinfolist.jsp
- 订单管理——orderlist.jsp
- 留言管理——noteslist.jsp
- 职工管理——employeelist.jsp
- 出版社管理——publisherlist.jsp

（2）Body

Body 是当前页面的主要内容显示区域。

（3）Bottom

Bottom 部分的文件名是 bottom.jsp，与客户端业务处理的 bottom.jsp 是同一个文件。使用＜％＠include file="../bottom.jsp"％＞语句嵌入管理端业务处理各个页面的底部，使得系统具有统一的风格。Bottom 部分主要用来显示版权信息、联系信息等。

10.2 用户登录功能实现

用户登录在客户端处理主界面的左侧，用户注册的信息存入 bookshop 数据库的 userinfo 表中。注册功能代码 login.jsp 由＜％＠include file="login.jsp" ％＞语句嵌入客户端主界面 index.jsp 文件。

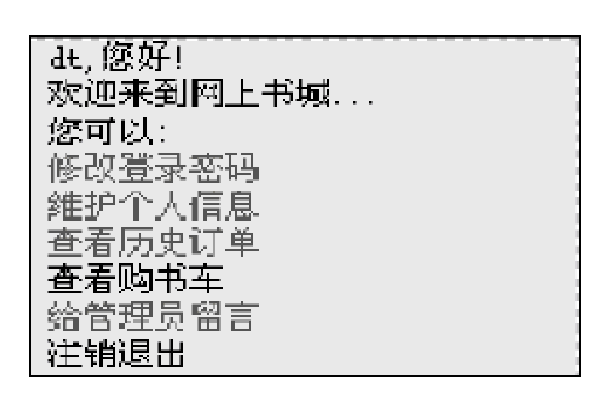
10.2.1 用户登录功能介绍

1. 老用户登录

用户登录界面见图 10-3(a)。如果是老用户，在文本框中输入用户名和密码，系统将把输入的用户名和密码与 bookshop 数据库中用户表 userinfo 中的信息进行比较验证。如果输入不正确，系统将提示用户并要求用户重新输入。若输入正确，则出现登录成功界面，如图 10-3(b)所示。



(a) 用户登录界面



(b) 登录成功界面

图 10-3 用户登录界面

2. 登录成功

用户登录成功后，可以在网上进行一系列的操作，它们的功能和完成这些功能的文件名如下：

- 修改登录密码——passwordedit.jsp
- 维护个人信息——userinfoedit.jsp
- 查看历史订单——myorder.jsp
- 查看购书车——shoppingcart.jsp
- 给管理员留言——leaveword.jsp
- 注销退出——注销退出

3. 新用户注册

如果是新用户，在注册界面单击“新用户注册”超链接，页面将跳转到“register.jsp”页面，需要用户输入个人信息，如图 10-4 所示。



图 10-4 新用户注册界面

把信息输入表单，单击“注册新用户”按钮，信息写入 bookshop 数据库的 userinfo 表中，并出现注册成功界面。返回主界面，即可以老用户身份登录。

10.2.2 用户登录界面 login.jsp 代码

```
<%@ page contentType="text/html; charset=GB2312"%>
<SCRIPT LANGUAGE="javascript">
<! --
function CheckSubmit()
```

```
 {
   if( document.loginform.userid.value=="")
      { alert("请输入用户名!");
        document.loginform.userid.focus(); return false; }
   if( document.loginform.password.value=="")
      { alert("请输入密码!");
        document.loginform.password.focus(); return false; }
   if(document.loginform.userid.value.indexOf("'") ! =-1)
      { alert("用户名不能包含单引号、空格等字符!");
        document.loginform.userid.focus(); return false; }
   return true;
  }
</SCRIPT>
<%@ page import="java.util.*,java.sql.*"%>
<%
   String op=request.getParameter("op");
   if(op!=null&&op.equals("login")){
      String s_userid=request.getParameter("userid");
      String s_password=request.getParameter("password");
      rs=stmt.executeQuery("select * from userinfo
          where userid='"+s_userid+"' and password='"+s_password+"'");
      if(rs.next()){
         session.setAttribute("userid",s_userid);
         response.sendRedirect("index.jsp");
      }
      else
      {
         response.sendRedirect("error.jsp? error="+"用户名或密码不正确!");
      }
   }
   if(op!=null&&op.equals("exit")){
      session.removeAttribute("userid");
      session.removeAttribute("cart");
      response.sendRedirect("index.jsp");
   }
   String s_userid=(String)session.getAttribute("userid");
%>
<%
   if(s_userid==null){
%>
<link href="maincss.css" rel="stylesheet" type="text/css">
<table width="100%" border="0" cellpadding="0"
       cellspacing="0" bgcolor="#f6f6f6" class="td">
   <form name="loginform" action="index.jsp? op=login" method="post">
```

```
        <tr>
          <td colspan="3"><div align="left">
         <img src="images/login_r1.jpg" width="158" height="43"></div></td>
        </tr>
        <tr>
          <td width="8"> </td>
          <td width="25%">用户名</td>
          <td width="75%">
         <input name="userid" type="text" class="formtext" size="12"></td>
        </tr>
        <tr>
          <td> </td>
          <td>密码</td>
          <td><input name="password" type="password"
                class="formtext" size="12"></td>
        </tr>
        <tr>
          <td> </td>
          <td colspan="2">
         <input name="loginbutton" type="submit" value="登录"
                    onClick="return CheckSubmit();">
               新用户<a href="register.jsp">注册</a></td>
        </tr>
      </form>
   </table>
   <%}else{%>
   <table width="100%" border="0" cellpadding="0"
             cellspacing="0" bgcolor="#f6f6f6" class="td">
      <tr>
        <td width="8"><div align="left"></div></td>
        <td width="737"><%=s_userid%>,您好!<br>
          欢迎来到网上书城…</td>
      </tr>
      <tr>
        <td> </td>
        <td>您可以:</td>
      </tr>
      <tr>
        <td> </td>
        <td><a href="passwordedit.jsp">修改登录密码</a></td>
      </tr>
      <tr>
        <td> </td>
        <td><a href="userinfoedit.jsp">维护个人信息</a></td>
```

```
    </tr>
    <tr>
      <td> </td>
      <td><a href="myorder.jsp">查看历史订单</a></td>
    </tr>
    <tr>
      <td> </td>
      <td><a href="shoppingcart.jsp">查看购书车</a></td>
    </tr>
    <tr>
      <td> </td>
      <td><a href="leaveword.jsp">给管理员留言</a></td>
    </tr>
    <tr>
      <td> </td>
      <td><a href="index.jsp? op=exit">注销退出</a></td>
    </tr>
</table>
<%}%>
```

10.2.3 新用户注册 register.jsp 代码

```
<%@ page contentType="text/html; charset=GB2312" errorPage="error.jsp"%>
<%@ page import="java.util.*,java.sql.*"%>
<SCRIPT LANGUAGE="javascript">
<!--
function CheckSubmit()
 {
    if(document.registerform.userid.value=="")
         { alert("请输入用户名!");
           document.registerform.userid.focus(); return false; }
    if(document.registerform.password.value=="")
         { alert("请输入密码!");
           document.registerform.password.focus(); return false; }
    if(document.registerform.password2.value=="")
         { alert("请验证密码!");
           document.registerform.password2.focus(); return false; }
    if(document.registerform.password2.value !=
         document.registerform.password.value )
         { alert("密码验证出错!");
           document.registerform.password.focus(); return false; }
    if(document.registerform.username.value=="")
         { alert("请输入真实姓名!");
```

```
            document.registerform.username.focus(); return false; }
    if(document.registerform.address.value=="")
        { alert("请输入住址!");
            document.registerform.address.focus(); return false; }
    if(document.registerform.postcode.value=="")
        { alert("请输入邮编!");
            document.registerform.postcode.focus(); return false; }
    if(document.registerform.phone.value=="")
        { alert("请输入联系电话!");
            document.registerform.phone.focus(); return false; }
    return true;
}
</SCRIPT>
<%
    request.setCharacterEncoding("GB2312");
    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
    Connection
    conn=DriverManager.getConnection("jdbc:odbc:bookshoplk","sa","");
    Statement stmt=conn.createStatement
          (ResultSet.TYPE_SCROLL_INSENSITIVE,ResultSet.CONCUR_READ_ONLY);
    ResultSet rs=null;
%>
<%
    String op=request.getParameter("op");
    if(op!=null&&op.equals("register")){
      String s_userid=request.getParameter("userid");
      String s_password=request.getParameter("password");
      String s_username=request.getParameter("username");
      String s_gender=request.getParameter("gender");
      String s_address=request.getParameter("address");
      String s_phone=request.getParameter("phone");
      String s_postcode=request.getParameter("postcode");
      String s_email=request.getParameter("email");
      String s_sql="insert into userinfo(userid,password,username,
                              gender,address,phone,postcode,email,state) ""
      values('"+s_userid+"','"+s_password+"','"+s_username+"','
      "+s_gender+"','"+s_address+"','"+s_phone+"','
      "+s_postcode+"','"+s_email+"','0')";
      try{
        stmt.executeUpdate(s_sql);
      }catch(Exception e){
        response.sendRedirect("error.jsp? error=注册失败,"+e.getMessage());
        }
  }
%>
<link href="maincss.css" rel="stylesheet" type="text/css">
```

```
<div align="center">
<table width="750" border="0" cellspacing="1" cellpadding="1">
  <tr>
    <td ><div align="center"><%@include file="top.jsp"%></div></td>
  </tr>
  <tr>
    <td ><div align="center">
  <table width="80%" border="0" cellpadding="1" cellspacing="1" class="td">
<%
  if(op==null){
%>
<form name="registerform" action="register.jsp? op=register" method="post">
    <tr>
      <td colspan="2">  以下资料请如实填写,以保证正确发货。
      这些资料未经客户允许将只用于处理客户的订货信息,对外严格保密。(带
        <font color="red">*</font>号的为必填项,其他的可以不填)</font></td>
    </tr>
    <tr>
      <td width="15%"><div align="right">用户名</div></td>
      <td><div align="left"><input name="userid" type="text" size="20">
        <font color="red">*</font>(为避免使用相同的用户名,请
        <a href="#">检测</a>你的用户名是否有效)</div></td>
    </tr>
    <tr>
      <td><div align="right">密码</div></td>
      <td><div align="left">
        <input name="password" type="password" size="20">
        <font color="red">*</font></div></td>
    </tr>
    <tr>
      <td><div align="right">验证密码</div></td>
      <td><div align="left">
        <input name="password2" type="password" size="20">
        <font color="red">*</font></div></td>
    </tr>
    <tr>
      <td><div align="right">真实姓名</div></td>
      <td><div align="left">
          <input name="username" type="text" size="20">
          <font color="red">*</font></div></td>
    </tr>
    <tr>
      <td><div align="right">性别</div></td>
      <td><div align="left">
         <select name="gender">
            <option value="男">男</option>
```

```
                <option value="女">女</option>
            </select>
        </div></td>
    </tr>
    <tr>
        <td><div align="right">住址</div></td>
        <td><div align="left">
            <input name="address" type="text" size="20">
            <font color="red"> * </font>(请您提供尽可能详细的地址)</div></td>
    </tr>
    <tr>
        <td><div align="right">邮编</div></td>
        <td><div align="left">
          <input name="postcode" type="text" size="20">
          <font color="red"> * </font></div></td>
    </tr>
    <tr>
        <td><div align="right">联系电话</div></td>
        <td><div align="left">
            <input name="phone" type="text" size="20">
            <font color="red"> * </font></div></td>
    </tr>
    <tr>
        <td><div align="right">E-mail</div></td>
        <td><div align="left">
            <input name="email" type="text" size="20"></div></td>
    </tr>
    <tr>
        <td><div align="right"> </div></td>
        <td><div align="left">
          <input name="submit" type="submit"
          value="注册新用户 " onClick="return CheckSubmit();">
          <input name="reset" type="reset" value="重新填写 ">
            </div></td>
    </tr>
</form>
<%}else{%>
    <tr>
          <td colspan="2">恭喜,注册成功,请记住您的用户名和密码! 现在就去
          <a href="index.jsp">首页登录</a>.</td>
    </tr>
<%}%>
   </table>
     </div>
    </td>
   </tr>
```

```
    <tr>
      <td ><div align="center"><%@include file="bottom.jsp"%></div></td>
    </tr>
  </table>
  </div>
```

10.3　图书展示功能实现

图书展示功能在网上书店客户端处理的主界面上，图书信息存在 bookshop 数据库的 book 表中。图书展示功能代码 search.jsp 由<%@include file="search.jsp"%>语句嵌入客户端主界面 index.jsp 文件。

10.3.1　图书展示功能介绍

1. 图书检索

图书检索界面见图 10-5。用户在文本框中输入需要检索的图书信息，单击“查找”按钮，页面将跳转到 booklist.jsp 页面，从数据库的 book 表中查询所需信息。如果没有找到需要查找的信息，系统将提示用户重新搜索；如果找到了，系统显示检索到的图书信息，如图 10-6 所示。

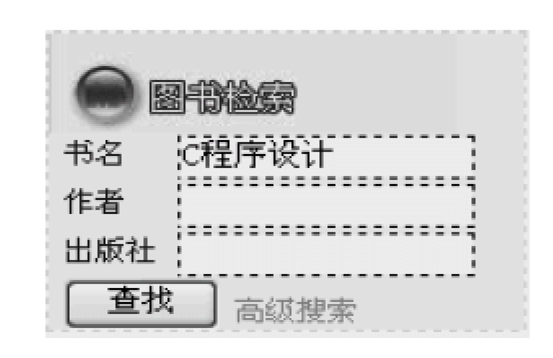


图 10-5　图书检索界面



图 10-6　书目搜索结果

2. 高级检索

在图 10-5 的界面上单击“高级搜索”超链接，页面将跳转至 booksearch.jsp，如图 10-7 所示。在该界面中输入需要检索的信息，单击“搜索”按钮，将跳转至 booklist.jsp 页面。

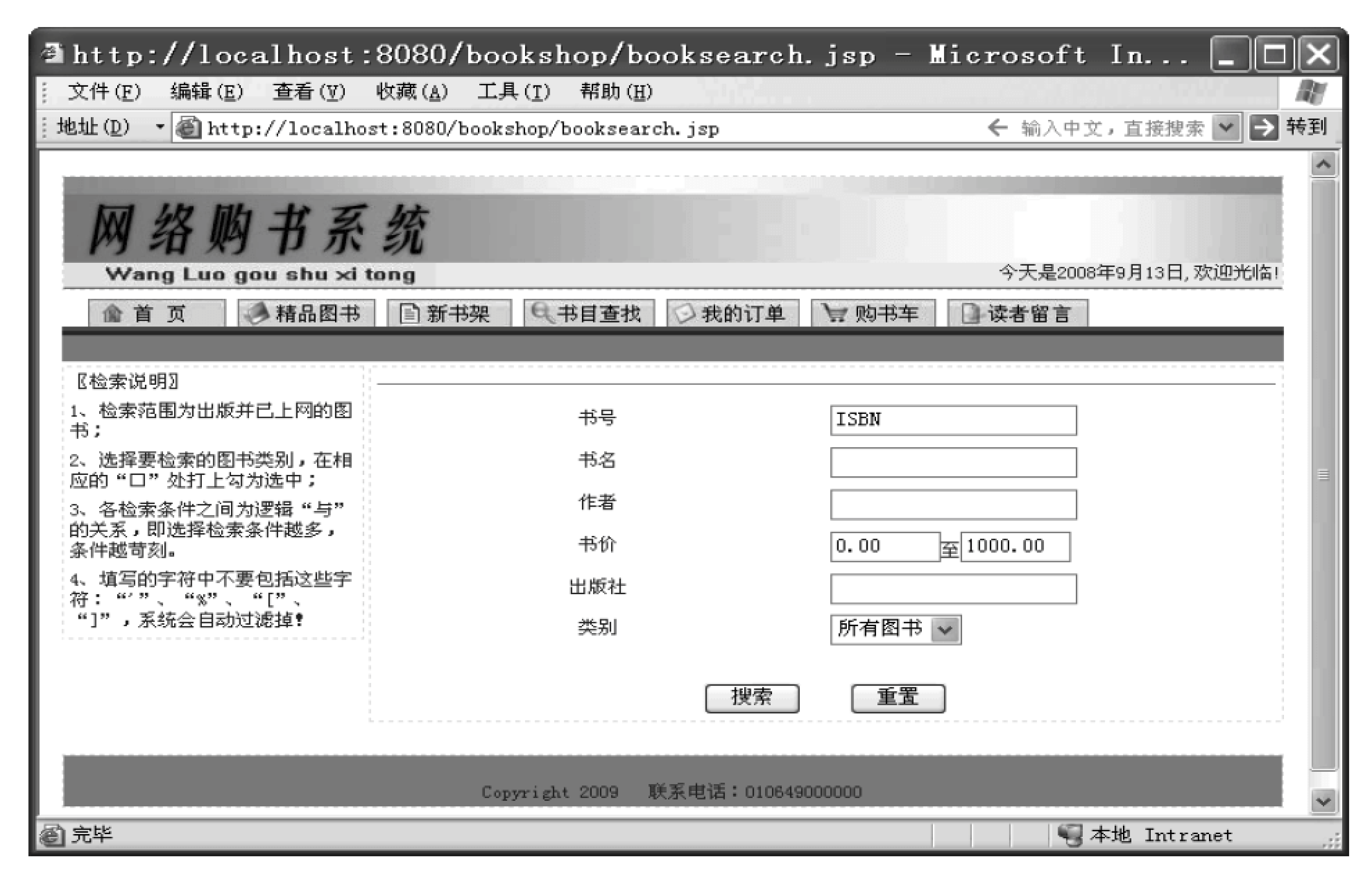


图 10-7　高级搜索

10.3.2　图书检索 search.jsp 代码

```
<%@ page contentType="text/html; charset=GB2312"%>
<%
  op=request.getParameter("op");
  if(op!=null&&op.equals("search")){
    String s_BookName=request.getParameter("bookname");
    String s_Author=request.getParameter("author");
    String s_Publisher=request.getParameter("publisher");
    String s_where="";
    if(!s_BookName.equals(""))
      s_where+="and bookname like'%"+s_BookName+"%' ";
    if(!s_Author.equals(""))
      s_where+="and author like'%"+s_Author+"%' ";
    if(!s_Publisher.equals(""))
      s_where+="and name like'%"+s_Publisher+"%' ";
    session.setAttribute("s_where",s_where);
    response.sendRedirect("booklist.jsp");
```

```
    }
%>
<link href="maincss.css" rel="stylesheet" type="text/css">
<table width="100%" border="0" cellpadding="0" cellspacing="0" bgcolor="#f6f6f6" class="td">
    <form name="searchform" action="index.jsp? op=search" method="post">
      <tr>
        <td colspan="3"><div align="left">
        <img src="images/search_r1.jpg" width="158" height="43"></div></td>
      </tr>
      <tr>
        <td width="8"> </td>
        <td width="25%">书名</td>
        <td width="75%"><input name="bookname"type="text"class="formtext"size="15"></td>
      </tr>
      <tr>
        <td> </td>
        <td>作者</td>
        <td><input name="author" type="text" class="formtext" size="15"></td>
      </tr>
      <tr>
        <td> </td>
        <td>出版社</td>
        <td><input name="publisher" type="text" class="formtext" size="15"></td>
      </tr>
      <tr>
        <td>  </td>
        <td colspan="2"><input name="submit" type="submit" value="查找">
            <a href="booksearch.jsp">高级搜索</a>
        </td>
      </tr>
    </form>
</table>
```

10.4 购书车实现

在网上书店浏览时，看见满意的图书，单击“放入购书车”超链接，即可将图书放入购书车。

10.4.1 购书车功能介绍

1. 相关文件名与数据库表

与购书车相关的代码名称与数据表名见表 10-1。

表 10-1　购书车代码表

功　能	代码名称	相关的数据库表名
将选中图书放入购书车	addtocart. jsp	图书表 book
显示购书车	shoppingcart. jsp	
购书数量加 1	increaseCart. jsp	
购书数量减 1	decreaseCart. jsp	
从购书车中取消某书目	delfromcart. jsp	
去收银台	order1. jsp	用户表 userinfo
下订单	order2. jsp	订单主表 orderform 订单子表 orderdetail
清空购书车	clearcart. jsp	

2. 购书车数据结构

购书车的数据结构应用 CartBean. java 代码，代码清单如下：

```
package bean;
public class cartBean {
  public String bookid="";
  public String bookname="";
  public String publish="";
  public int ordernum=0;
  public double unitprice=0.0;
  public double subtotal=0.0;
}
```

变量含义：

bookid:　　书号
bookname:　书名
publish:　　出版社
ordernum:　订购数量
unitprice:　单价
subtotal:　合价

3. 显示购书车

在网上选中图书后，单击"放入购书车"超链接，调用 addtocart. jsp 代码，将选中的图书放入购书车，再跳转至 shoppingcart. jsp，显示购书车的内容，见图 10-8。

在该界面可以完成以下操作：

① 单击"取消"超链接，调用 delfromcart. jsp 代码将该书目从购书车中删除。



图 10-8　显示购书车

② 单击“清空购书车”按钮，调用 clearcart.jsp 代码，把购书车清空。

③ 单击“继续购书”按钮，返回客户端处理主界面 index.jsp 代码，继续购书。

④ 单击“＋”或“－”按钮，调用 increaseCart.jsp 或 decreaseCart.jsp 代码，更改数量。

⑤ 单击“去收银台”按钮，调用 order1.jsp 代码，跳转至图 10-9 下订单界面。



图 10-9　下订单界面

4. 下订单

在图 10-9 的下订单界面单击“我要下订单”按钮，页面跳转到 order2.jsp，显示订单信息，如图 10-10 所示。



图 10-10　订单

5. 我的订单

在网上书店的客户端处理界面上方单击“我的订单”按钮，调用 myorder.jsp 页面，显示该读者的全部订单信息，如图 10-11 所示。

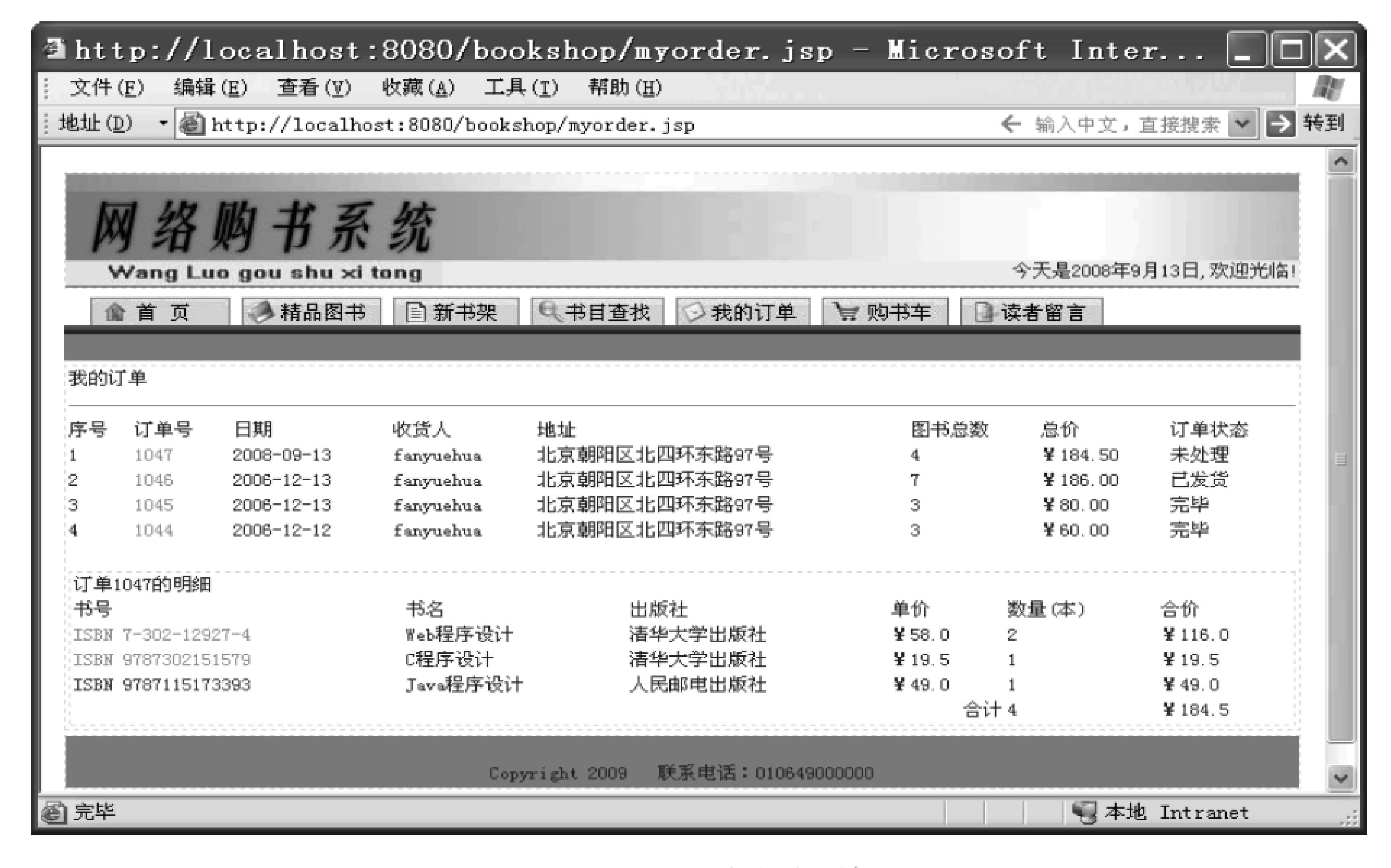


图 10-11　我的订单

10.4.2　放入购书车 addtocart.jsp 代码

```
<%@ page contentType="text/html; charset=GB2312"%>
<%@ page import="bean.*,java.util.*"%>
```

```
<%@ page import="java.sql.*"%>
<%
    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
    Connection conn=DriverManager.getConnection("jdbc:odbc:bookshoplk","sa","");
    Statement stmt=conn.createStatement
                (ResultSet.TYPE_SCROLL_INSENSITIVE,ResultSet.CONCUR_READ_ONLY);
    ResultSet rs=null;
%>
<%
    String s_bookid=request.getParameter("bookid");
    rs=stmt.executeQuery("select * from book  left join publisher on
        book.publisherid=publisher.publisherid   where bookid='"+s_bookid+"'");
    rs.next();
    String s_bookname=rs.getString("bookname");
    String s_author=rs.getString("author");
    String s_price=rs.getString("price");
    String s_publish=rs.getString("name");
    int i_ordernum=1;
    double d_unitprice=Double.parseDouble(s_price);
    double d_subtotal=i_ordernum * d_unitprice;
    d_subtotal=Math.round(d_subtotal * 100)/100.00;
    cartBean cbean=new cartBean();
    cbean.bookid=s_bookid;
    cbean.bookname=s_bookname;
    cbean.publish=s_publish;
    cbean.ordernum=i_ordernum;
    cbean.unitprice=d_unitprice;
    cbean.subtotal=d_subtotal;
    Collection c_cart=(Collection)session.getAttribute("cart");
    Iterator it=null;
%>
<%
    if(c_cart==null){
      c_cart=new Vector();
      c_cart.add(cbean);
      session.setAttribute("cart",c_cart);
    }else{
      String s_flag="false";
      it=c_cart.iterator();
      while(it.hasNext()){
        cartBean cb=(cartBean)(it.next());
        if(cb.bookid.equals(s_bookid)){
cb.ordernum++;
            cb.subtotal+=cb.unitprice;
s_flag="true";
        }//if end
```

```
        }//while end
        if(s_flag.equals("false")){
          c_cart.add(cbean);
        }
      }//else end
      response.sendRedirect("shoppingcart.jsp");
%>
```

10.4.3 显示购书车 shoppingcart.jsp 代码

```
<%@ page contentType="text/html; charset=GB2312"%>
<%@ page import="bean.*,java.util.*"%>
<script language="JavaScript" type="text/JavaScript">
<!--
   function MM_goToURL() { //v3.0
      var i, args=MM_goToURL.arguments; document.MM_returnValue=false;
      for (i=0; i<(args.length-1); i+=2) eval(args[i]+".location='"+args[i+1]+"'");
}
//-->
</script>
<link href="maincss.css" rel="stylesheet" type="text/css">
<div align="center">
<table width="750" border="0" cellspacing="1" cellpadding="1">
  <tr>
    <td ><div align="center"><%@include file="top.jsp"%></div></td>
  </tr>
  <tr>
    <td ><div align="center">
      <table width="100%" border="0" cellpadding="1" cellspacing="1" class="td">
        <tr>
          <td colspan="7">购书车</td>
        </tr>
        <tr>
          <td colspan="7"><hr size="1" noshade width="100%"></td>
        </tr>
<%
  int i_totalnum=0;
  double d_totalamount=0.0;
  Collection c_cart=(Collection)session.getAttribute("cart");
  Iterator it=null;
  if(c_cart!=null&&c_cart.size()>0){%>
        <tr>
          <td>书号</td>
          <td>书名</td>
```

```
            <td>出版社</td>
            <td>单价</td>
            <td>数量(本)</td>
            <td>合价</td>
    <td>操作</td>
          </tr>
<%
    it=c_cart.iterator();
    while(it.hasNext()){
      cartBean cbean=(cartBean)(it.next());
      i_totalnum+=cbean.ordernum;
      d_totalamount+=cbean.subtotal;
%>
          <tr>
            <td height="19"><a href="bookdetail.jsp? bookid=<%=cbean.bookid%>"
              target="_blank"><%=cbean.bookid%></a></td>
            <td><%=cbean.bookname%></td>
            <td><%=cbean.publish%></td>
            <td>¥<%=cbean.unitprice%></td>
            <td><input type="text" class="formtext" value="<%=cbean.ordernum%>
                "size="5" readonly> 
        <input type="button" onClick="MM_goToURL('parent','increaseCart.jsp?
                bookid=<%=cbean.bookid%>');return document.MM_returnValue" value=
                "+"> 
        <input type="button" onClick="MM_goToURL('parent','decreaseCart.jsp?
                bookid=<%=cbean.bookid%>');return document.MM_returnValue" value=
                "-"></td>
            <td>¥<%=cbean.subtotal%></td>
     <td><a href="delfromcart.jsp? bookid=<%=cbean.bookid%>">取消</a></td>
          </tr>
<%   }//while%>
          <tr>
            <td colspan="4"><div align="right">合计</div></td>
            <td ><%=i_totalnum%>(本)</td>
<%
    d_totalamount=Math.round(d_totalamount * 100)/100.00;
%>
            <td colspan="2">¥<%=d_totalamount%></td>
          </tr>
          <tr>
            <td colspan="7"> <div align="center">
              <input type="button" onClick="MM_goToURL('parent','clearcart.jsp');
                  return document.MM_returnValue" value="清空购书车">   
              <input type="button" onClick="MM_goToURL('parent','index.jsp');
                  return document.MM_returnValue" value="继续购书">  
              <input type="button" onClick="MM_goToURL('parent','order1.jsp');
```

```
                return document.MM_returnValue" value="去收银台">
              </div></td>
        </tr>
<%}else{%>
        <tr>
          <td colspan="7"><div align="center">您的购书车为空！</div></td>
        </tr>
<% }//else%>
      </table>
    </div></td>
  </tr>
  <tr>
    <td><div align="center"><%@include file="bottom.jsp"%></div></td>
  </tr>
</table>
</div>
```

10.5 读者留言功能实现

10.5.1 读者留言功能介绍

读者成功登录网上书店后，单击页面左方"给管理员留言"超链接，进入leaveword.jsp留言界面，显示读者留言，见图10-12。读者留言后，单击"留言"按钮，



图10-12 读者留言

调用leaveword2.jsp代码，将留言写入 nodes 表，并返回 leaveword.jsp 显示留言。

10.5.2 读者留言 leaveword.jsp 代码

```
<%@ page contentType="text/html; charset=GB2312"%>
<%@ page import="java.util.*,java.sql.*"%>
<SCRIPT LANGUAGE="javascript">
<!--
function CheckSubmit()
 {
  if( document.form.subject.value=="")
     { alert("请输入主题!"); document.form.subject.focus(); return false; }
  if( document.form.context.value=="")
     { alert("请输入内容!"); document.form.context.focus(); return false; }
  if( document.form.context.value.length >=100 )
     { alert("留言内容不能大于 100 字!"); document.form.context.focus(); return false; }
  return true;
}
</SCRIPT>
<%
  request.setCharacterEncoding("GB2312");
  Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
  Connection conn=DriverManager.getConnection("jdbc:odbc:bookshoplk","sa","");
  Statement stmt=conn.createStatement
                (ResultSet.TYPE_SCROLL_INSENSITIVE,ResultSet.CONCUR_READ_ONLY);
  ResultSet rs=null;
  String s_userid=(String)session.getAttribute("userid");
  int i_totalnum=0;          //总记录数
  int i_pagenum=5;           //一页显示的记录数码
  int i_totalpage=0;         //总页数
  int i_currentpage=1;       //当前页
  String s_createid="";
  String s_subject="";
  String s_date1="";
  String s_context="";
  String s_employeeid="";
  String s_date2="";
  String s_advice="";
  rs=stmt.executeQuery("select count( * ) from notes  ");
  rs.next();
  i_totalnum=rs.getInt(1);
  i_totalpage=i_totalnum/i_pagenum;
  if(i_totalnum%i_pagenum!=0)
    i_totalpage++;
  String op=request.getParameter("op");
```

```
    if(op!=null&&op.equals("show")){
      String s_currentpage=request.getParameter("currentpage");
      i_currentpage=Integer.parseInt(s_currentpage);
    }
    if(op!=null&&op.equals("previous")){
      String s_currentpage=request.getParameter("currentpage");
      i_currentpage=Integer.parseInt(s_currentpage)-1;
    }
    if(op!=null&&op.equals("next")){
      String s_currentpage=request.getParameter("currentpage");
      i_currentpage=Integer.parseInt(s_currentpage)+1;
    }
    if(op!=null&&op.equals("firstpage")){
      i_currentpage=1;
    }
    if(op!=null&&op.equals("lastpage")){
      i_currentpage=i_totalpage;
    }
%>
<link href="maincss.css" rel="stylesheet" type="text/css">
<div align="center">
<table width="750" border="0" cellspacing="1" cellpadding="1">
  <tr>
    <td ><div align="center"><%@include file="top.jsp"%></div></td>
  </tr>
  <tr>
    <td ><div align="center">
      <table width="100%" border="0" cellpadding="1" cellspacing="1" class="td">
        <tr>
          <td colspan="6">读者留言</td>
        </tr>
        <tr>
          <td colspan="6"><hr size="1" noshade width="100%"></td>
        </tr>
<%if(i_totalnum!=0){%>
        <tr>
          <td colspan="3">最近的留言共<%=i_totalnum%>条,分<%=i_totalpage%>
                    页显示,每页显示<%=i_pagenum%>条</td>
          <td colspan="3"><div align="right">
            第<%=i_currentpage%>页  
              <% if(i_currentpage>1){%>
              <a href="leaveword.jsp? op=firstpage">首页</a>
              <% }else{%>首页<%}%> 
              <% if(i_currentpage>1){%>
                <a href="leaveword.jsp? op=previous&currentpage=<%=i_currentpage%
                >">上一页</a>
```

```
                <% }else{%>上一页<%}%> 
                <% if(i_currentpage!=i_totalpage){%>
                  <a href="leaveword.jsp? op=next&currentpage=<%=i_currentpage%>"
                  >下一页</a>
                <% }else{%>下一页<%}%> 
                  <% if(i_currentpage!=i_totalpage){%>
                    <a href="leaveword.jsp? op=lastpage">末页</a>
                  <% }else{%>末页<%}%>
              </div></td>
          </tr>
          <tr>
            <td colspan="6"> </td>
          </tr>
  <tr>
      <td width="10%">留言日期</td>
      <td width="10%">留言人</td>
      <td width="15%">主题</td>
      <td width="25%">内容</td>
      <td width="10%">处理日期</td>
      <td width="30%">处理意见</td>
  </tr>
<%
  rs=stmt.executeQuery("select * from notes   order by id desc");
  if(i_currentpage>i_totalpage)
  i_currentpage=i_totalpage;
  int i_position=(i_currentpage-1) * i_pagenum;
  if(i_position==0)
    rs.beforeFirst();
  else
    rs.absolute(i_position);
  for(int i=0;i<i_pagenum;i++){
    if(!rs.isLast()){
      rs.next();
      s_createid=(rs.getString("userid")!=null?rs.getString("userid"):"");
      s_subject=(rs.getString("subject")!=null?rs.getString("subject"):"");
      s_date1=(rs.getString("date1")!=null?rs.getString("date1").substring(0,10):"");
      s_context=(rs.getString("context")!=null?rs.getString("context"):"");
      s_employeeid=(rs.getString("employeeid")!=null?rs.getString("employeeid"):"");
      s_date2=(rs.getString("date2")!=null?rs.getString("date2").substring(0,10):"");
      s_advice=(rs.getString("advice")!=null?rs.getString("advice"): "[请等待工作人员处理]");
%>
<tr>
      <td ><%=s_date1%></td>
      <td ><%=s_createid%></td>
      <td ><%=s_subject%></td>
      <td ><%=s_context%></td>
```

```
        <td ><%=s_date2%></td>
        <td ><%=s_advice%></td>
</tr>
<%
        }//if
    }//for
%>
<%}else{%>
            <tr>
              <td colspan="6"> </td>
            </tr>
<%}%>
<%
  if(s_userid!=null){
%>
<form name="form" action="leaveword2.jsp" method="post">
            <tr>
              <td width="9%">主题</td>
              <td colspan="5"><input name="subject" type="text" size="25"></td>
            </tr>
            <tr>
              <td width="9%">留言</td>
              <td colspan="5"><textarea name="context" cols="70" rows="10"></textarea></td>
            </tr>
            <tr>
              <td colspan="6"><div align="center"><input name="userid"
                                          type="hidden" value="<%=s_userid%>">
                <input name="submit" type="submit" value="留言" onClick="return CheckSubmit();">
                  </div></td>
            </tr>
</form>
<%}else{%>
            <tr>
              <td colspan="6"> </td>
            </tr>
            <tr>
              <td colspan="6">您还没有登录,登录用户可以发表留言!</td>
            </tr>
<%}%>
        </table></div></td>
   </tr>
   <tr>
     <td><div align="center"><%@include file="bottom.jsp"%></div></td>
   </tr>
</table>
</div>
```

10.5.3　将留言写入数据库 leaveword2.jsp 代码

```
<%@ page contentType="text/html; charset=GB2312"%>
<%@ page import="java.util.*,java.sql.*"%>
<%
    request.setCharacterEncoding("GB2312");
    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
    Connection conn=DriverManager.getConnection("jdbc:odbc:bookshoplk","sa","");
    Statement stmt=conn.createStatement
                (ResultSet.TYPE_SCROLL_INSENSITIVE,ResultSet.CONCUR_READ_ONLY);
    ResultSet rs=null;
    String s_userid=request.getParameter("userid");
    String s_subject=request.getParameter("subject");
    String s_context=request.getParameter("context");
    String s_sql="insert into notes(userid,subject,date1,context) "+
              "values('"+s_userid+"','"+s_subject+"',getdate(),'"+s_context+"')";
    stmt.executeUpdate(s_sql);
    response.sendRedirect("leaveword.jsp");
%>
```

10.6　订单管理功能实现

管理端业务处理主要有：图书管理、用户管理、订单管理、留言管理、出版社管理和职工管理等模块，以订单管理为例说明它们的实现方法。

10.6.1　订单管理功能介绍

发布在 bookshop/admin 目录下的 index.jsp 文件，打开管理端业务处理主界面，见图 10-13。管理员在界面中输入用户名和密码，系统将此密码和用户名与数据库中职工表 employee 中的信息对照，如果输入不正确，系统将提示管理员重新输入；若输入正确，将转至管理员界面。管理员单击“订单管理”按钮，调用 orderlist.jsp 显示数据库订单表 orderform 中的数据，见图 10-14。



图 10-13　管理员身份验证

订单有三种状态：未处理、已发货和完毕。下订单后的订单是“未处理”状态。管理员发货后，将订单状态改为“已发货”。送货员从用户处收回货款后，将状态改为“完毕”。管理员可以在显示订单页面上更改订单状



图 10-14　显示订单

态，单击页面上“更改状态”栏目的超链接，调用 orderedit.jsp 代码改变订单的状态。管理员也可以单击页面右上角的“查询”超链接，弹出订单查询界面 ordersearch.jsp，查找某一订单，并把它显示出来，如图 10-15 所示。



图 10-15　订单查询

管理员输入查询条件后，单击“查询”按钮，调用 orderlist.jsp 显示需要查询的订单。

10.6.2 管理员身份验证 bookshop/admin/index.jsp 代码

```
<%@ page contentType="text/html; charset=GB2312"%>
<%@ page import="java.util.*,java.sql.*"%>
<SCRIPT LANGUAGE="javascript">
<!--
function CheckSubmit()
 {
  if( document.loginform.employeeid.value=="")
     { alert("请输入用户名!"); document.loginform.employeeid.focus(); return false; }
  if( document.loginform.password.value=="")
     { alert("请输入密码!"); document.loginform.password.focus(); return false; }
  if(document.loginform.employeeid.value.indexOf("'")
                                   !=-1||document.loginform.employeeid.value.indexOf("") !=-1)
     { alert("用户名不能包含单引号、空格等字符!"); document.loginform.employeeid.focus();
       return false; }
  return true;
 }
</SCRIPT>
<%
  request.setCharacterEncoding("GB2312");
  Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
  Connection conn=DriverManager.getConnection("jdbc:odbc:bookshoplk","sa","");
  Statement stmt=conn.createStatement
              (ResultSet.TYPE_SCROLL_INSENSITIVE,ResultSet.CONCUR_READ_ONLY);
  ResultSet rs=null;
  String op=request.getParameter("op");
  if(op!=null&&op.equals("login")){
    String s_employeeid=request.getParameter("employeeid");
    String s_password=request.getParameter("password");
    rs=stmt.executeQuery("select * from employee where
                employeeid='"+s_employeeid+"' and password='"+s_password+"'");
    if(rs.next()){
      session.setAttribute("admin",s_employeeid);
      response.sendRedirect("booklist.jsp");
    }
    else{
      response.sendRedirect("../error.jsp? error="+"用户名或密码不正确!");
    }
  }
%>
<link href="../maincss.css" rel="stylesheet" type="text/css">
```

```
<div align="center">
<br><br><br><br>
  <table width="20%" border="0" cellpadding="0" cellspacing="0" class="td">
  <form name="loginform" action="index.jsp? op=login" method="post">
    <tr>
      <td colspan="2"><img src="../images/adin_r1.jpg"></td>
    </tr>
    <tr>
      <td width="25%">用户名</td>
      <td><input name="employeeid" type="text"  size="15"></td>
    </tr>
    <tr>
      <td>密码</td>
      <td><input name="password" type="password"  size="15"></td>
    </tr>
    <tr>
      <td colspan="2"><div align="center">
          <input name="submit" type="submit"  onClick="return CheckSubmit();" value="登录">
        </div></td>
    </tr>
 </form>
  </table>
</div>
```

10.6.3 订单处理 orderedit.jsp 代码

```
<%@ page contentType="text/html; charset=GB2312"%>
<%@ page import="java.util.*,java.sql.*"%>
<%
  Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
  Connection conn=DriverManager.getConnection("jdbc:odbc:bookshoplk","sa","");
  Statement stmt=
     conn.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,ResultSet.CONCUR_
     READ_ONLY);
  ResultSet rs=null;
  String s_admin=(String)session.getAttribute("admin");
  if(s_admin==null){
    response.sendRedirect("checklogin.jsp");
  }
  String s_currentpage=request.getParameter("currentpage");
  String s_orderid=request.getParameter("orderid");
  String s_state=request.getParameter("state");
  stmt.executeUpdate("update orderform set state='"+s_state+"' where orderid='"+s_orderid+"'");
```

```
  rs=stmt.executeQuery("select * from orderdetail where orderid='"+s_orderid+"'");
  while(rs.next()){
    String s_bookid=rs.getString("bookid");
    int i_ordernum=rs.getInt("ordernum");
    Statement stmt2=conn.createStatement
            (ResultSet.TYPE_SCROLL_INSENSITIVE,ResultSet.CONCUR_READ_ONLY);
    stmt2.executeUpdate("update book set quantity=quantity-"+i_ordernum+"where bookid
    ='"+s_bookid+"'");
  }
  response.sendRedirect("orderlist.jsp? op=show&currentpage="+s_currentpage);
%>
```

上机练习与实训 10

一、上机练习

1. 完成留言板的制作，界面如图 10-16 所示。



图 10-16 留言板

2. 完成一个网上聊天室的制作。

二、实训课题

1. 完成仓库内部管理系统的分析、设计与实现。主界面如图 10-17 所示。要求具有：显示商品、添加商品、商品入库、商品出库、查找商品和清除商品等功能。

2. 完成 BBS 论坛的制作，包含有以下功能：

(1) 会员管理子系统：实现会员的在线注册、登录时身份验证、个人信息修改。

(2) 文章管理子系统：显示文章主题、阅读文章、发表文章和跟帖文章。

(3) 留言管理：提交留言、浏览留言和回复留言。

3. 完成图书馆管理信息系统的制作，系统功能见图 10-18。

4. 完成网上考试系统的制作，系统功能如下：

(1) 身份验证子系统

只有被授权的用户才可以登录考试系统，本系统把登录系统的身份定为三种，不同的



图 10-17　仓库内部管理系统

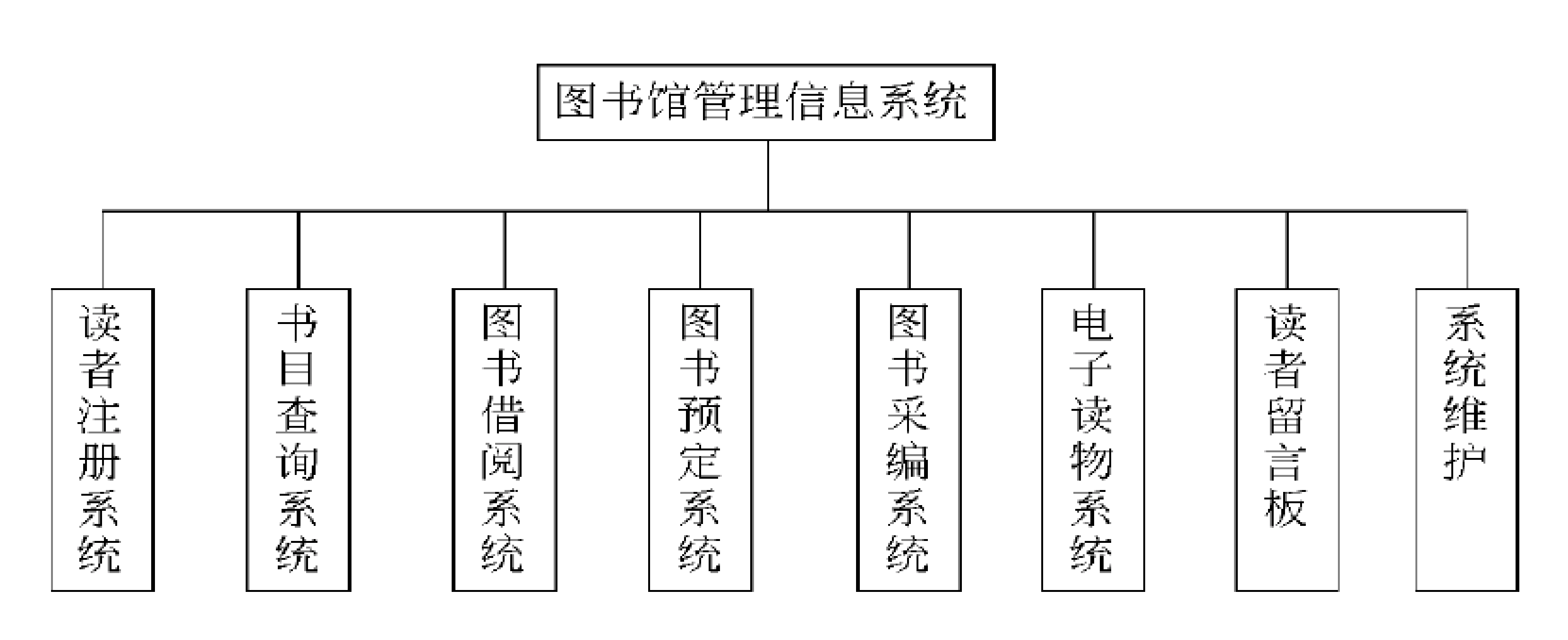


图 10-18　图书馆管理信息系统功能

身份具有不同的权限：

- 高级管理员：具有整个系统的应用权限。
- 管理员：可以使用题库制作子系统和成绩管理子系统。
- 考生：参加考试。

（2）系统管理子系统。

（3）考生管理子系统。

（4）题库管理子系统。

（5）监控中心子系统。

附录 A　网上资源使用说明

本书各章的例题全部调试通过,按章节存放在清华大学出版社网站上本书相应信息中。使用方法如下。

1. 第 2 章案例使用

例 2.1 ex2-01.jsp,把 ex_D02 目录下的 ex2-01.jsp 文件存放在\Tomcat 5.5\webapps\ex_D02 目录下。

2. 第 2 篇案例使用

第 2 篇的案例存放在 ex_D04、ex_D05 和 ex_D06 目录下,双击文件名即可执行。

3. 第 3 篇案例使用

第 3 篇所有案例在"Windows XP + Tomcat 5.5 + SQL 个人版"环境上调试通过。服务器端的应用程序应放在 Web 的发布目录下或其虚拟路径下。本书第 3 篇的案例应存放在\tomcat 5.5\webapps 下。

第 7 章、第 8 章和第 9 章的所有案例存放在 ex_D07、ex_D08 和 ex_D09 目录下,直接把 ex_D07、ex_D08 或 ex_D09 文件夹拖到\tomcat 5.5\webapps\ 目录下,例如\tomcat 5.5\webapps\ ex_D07,即可发布运行。

4. 网上书店的安装及使用

为使读者在学习过程中易于理解,并参考使用,将系统适当剪裁并移植到计算机环境中,读者可以将网上 bookshop 中的内容复制到 Tomcat 的发布目录下即可运行。

安装操作步骤如下:

① 安装 jdk1.6.0。

② 安装 Tomcat 5.5。

③ 安装 MS SQL Server 2000。

④ 将 bookshop 下的数据库备份文件 bookshop804 导入到数据库中,数据库取名为 bookshop。把数据库备份文件 bookshop804 还原到读者系统中的操作如下:

- 在桌面上选择"开始→所有程序→Microsoft SQL Server→企业管理器",进入数据库企业管理器窗口。
- 在"企业管理器"窗口中用鼠标右击"数据库",选择"所有任务→还原数据库"。
- 显示"还原为数据库"窗口,在"还原为数据库"文本框中输入数据库名称 bookshop;选择"从设备"单选钮,弹出"参数框";在参数框中单击"选择设备"按钮。
- 系统显示"选择还原设备"窗口,单击"添加"按钮,进入"选择还原目的"窗口。

- 在“选择还原目的”窗口中浏览数据库备份文件(文件名为 bookshop804),单击“确定”按钮,系统自动将 bookshop 数据库还原到读者系统中。

⑤ 创建数据源,选择“控制面板→管理工具→数据源”建立与该数据库对应的数据源,数据源取名为 bookshoplk。

⑥ 将 bookshop 整个文件夹复制到 tomcat 发布目录 webapps 下。

⑦ 以上步骤完成后,重启 Tomcat,在浏览器地址栏目中输入:

http://localhost:8080/bookshop,客户端(初始账号和密码:pds/pass)。

http://localhost:8080/bookshop/admin,管理端(初始账号和密码:admin/pass)。

进入网上书店。

高等学校计算机基础教育教材精选

书　名	书　号
Access 数据库基础教程　赵乃真	ISBN 978-7-302-12950-9
AutoCAD 2002 实用教程　唐嘉平	ISBN 978-7-302-05562-4
AutoCAD 2006 实用教程(第 2 版)　唐嘉平	ISBN 978-7-302-13603-3
AutoCAD 2007 中文版机械制图实例教程　蒋晓	ISBN 978-7-302-14965-1
AutoCAD 计算机绘图教程　李苏红	ISBN 978-7-302-10247-2
C++ 及 Windows 可视化程序设计　刘振安	ISBN 978-7-302-06786-3
C++ 及 Windows 可视化程序设计题解与实验指导　刘振安	ISBN 978-7-302-09409-8
C++ 语言基础教程(第 2 版)　吕凤翥	ISBN 978-7-302-13015-4
C++ 语言基础教程题解与上机指导(第 2 版)　吕凤翥	ISBN 978-7-302-15200-2
C++ 语言简明教程　吕凤翥	ISBN 978-7-302-15553-9
CATIA 实用教程　李学志	ISBN 978-7-302-07891-3
C 程序设计教程(第 2 版)　崔武子	ISBN 978-7-302-14955-2
C 程序设计辅导与实训　崔武子	ISBN 978-7-302-07674-2
C 程序设计试题精选　崔武子	ISBN 978-7-302-10760-6
C 语言程序设计　牛志成	ISBN 978-7-302-16562-0
PowerBuilder 数据库应用系统开发教程　崔巍	ISBN 978-7-302-10501-5
Pro/ENGINEER 基础建模与运动仿真教程　孙进平	ISBN 978-7-302-16145-5
SAS 编程技术教程　朱世武	ISBN 978-7-302-15949-0
SQL Server 2000 实用教程　范立南	ISBN 978-7-302-07937-8
Visual Basic 6.0 程序设计实用教程(第 2 版)　罗朝盛	ISBN 978-7-302-16153-0
Visual Basic 程序设计实验指导与习题　罗朝盛	ISBN 978-7-302-07796-1
Visual Basic 程序设计教程　刘天惠	ISBN 978-7-302-12435-1
Visual Basic 程序设计应用教程　王瑾德	ISBN 978-7-302-15602-4
Visual Basic 试题解析与实验指导　王瑾德	ISBN 978-7-302-15520-1
Visual Basic 数据库应用开发教程　徐安东	ISBN 978-7-302-13479-4
Visual C++ 6.0 实用教程(第 2 版)　杨永国	ISBN 978-7-302-15487-7
Visual FoxPro 程序设计　罗淑英	ISBN 978-7-302-13548-7
Visual FoxPro 数据库及面向对象程序设计基础　宋长龙	ISBN 978-7-302-15763-2
Visual LISP 程序设计(AutoCAD 2006)　李学志	ISBN 978-7-302-11924-1
Web 数据库技术　铁军	ISBN 978-7-302-08260-6
Web 技术应用基础(第 2 版)　樊月华等	ISBN 978-7-302-18840-7
程序设计教程(Delphi)　姚普选	ISBN 978-7-302-08028-2
程序设计教程(Visual C++)　姚普选	ISBN 978-7-302-11134-4
大学计算机(应用基础·Windows 2000 环境)卢湘鸿	ISBN 978-7-302-10187-1
大学计算机基础　高敬阳	ISBN 978-7-302-11566-3
大学计算机基础实验指导　高敬阳	ISBN 978-7-302-11545-8
大学计算机基础　秦光洁	ISBN 978-7-302-15730-4
大学计算机基础实验指导与习题集　秦光洁	ISBN 978-7-302-16072-4
大学计算机基础　牛志成	ISBN 978-7-302-15485-3
大学计算机基础　訾秀玲	ISBN 978-7-302-13134-2
大学计算机基础习题与实验指导　訾秀玲	ISBN 978-7-302-14957-6
大学计算机基础教程(第 2 版)　张莉	ISBN 978-7-302-15953-7
大学计算机基础实验教程(第 2 版)　张莉	ISBN 978-7-302-16133-2
大学计算机基础实践教程(第 2 版)　王行恒	ISBN 978-7-302-18320-4
大学计算机技术应用　陈志云	ISBN 978-7-302-15641-3

大学计算机软件应用　王行恒　ISBN 978-7-302-14802-9
大学计算机应用基础　高光来　ISBN 978-7-302-13774-0
大学计算机应用基础上机指导与习题集　郝莉　ISBN 978-7-302-15495-2
大学计算机应用基础　王志强　ISBN 978-7-302-11790-2
大学计算机应用基础题解与实验指导　王志强　ISBN 978-7-302-11833-6
大学计算机应用基础教程　詹国华　ISBN 978-7-302-11483-3
大学计算机应用基础实验教程(修订版)　詹国华　ISBN 978-7-302-16070-0
大学计算机应用教程　韩文峰　ISBN 978-7-302-11805-3
大学信息技术(Linux 操作系统及其应用)　衷克定　ISBN 978-7-302-10558-9
电子商务网站建设教程(第二版)　赵祖荫　ISBN 978-7-302-16370-1
电子商务网站建设实验指导　赵祖荫　ISBN 978-7-302-07941-5
多媒体技术及应用　王志强　ISBN 978-7-302-08183-8
多媒体技术及应用　付先平　ISBN 978-7-302-14831-9
多媒体应用与开发基础　史济民　ISBN 978-7-302-07018-4
基于 Linux 环境的计算机基础教程　吴华洋　ISBN 978-7-302-13547-0
基于开放平台的网页设计与编程(第 2 版)　程向前　ISBN 978-7-302-18377-8
计算机辅助工程制图　孙力红　ISBN 978-7-302-11236-5
计算机辅助设计与绘图(AutoCAD 2007 中文版)(第 2 版)　李学志　ISBN 978-7-302-15951-3
计算机软件技术及应用基础　冯萍　ISBN 978-7-302-07905-7
计算机图形图像处理技术与应用　何薇　ISBN 978-7-302-15676-5
计算机网络公共基础　史济民　ISBN 978-7-302-05358-3
计算机网络基础(第 2 版)　杨云江　ISBN 978-7-302-16107-3
计算机网络技术与设备　满文庆　ISBN 978-7-302-08351-1
计算机文化基础教程(第 2 版)　冯博琴　ISBN 978-7-302-10024-9
计算机文化基础教程实验指导与习题解答　冯博琴　ISBN 978-7-302-09637-5
计算机信息技术基础教程　杨平　ISBN 978-7-302-07108-2
计算机应用基础　林冬梅　ISBN 978-7-302-12282-1
计算机应用基础实验指导与题集　冉清　ISBN 978-7-302-12930-1
计算机应用基础题解与模拟试卷　徐士良　ISBN 978-7-302-14191-4
计算机应用基础教程　姜继忱　徐敦波　ISBN 978-7-302-18421-8
计算机硬件技术基础　李继灿　ISBN 978-7-302-14491-5
软件技术与程序设计(Visual FoxPro 版)　刘玉萍　ISBN 978-7-302-13317-9
数据库应用程序设计基础教程(Visual FoxPro)　周山芙　ISBN 978-7-302-09052-6
数据库应用程序设计基础教程(Visual FoxPro)题解与实验指导　黄京莲　ISBN 978-7-302-11710-0
数据库原理及应用(Access)(第 2 版)　姚普选　ISBN 978-7-302-13131-1
数据库原理及应用(Access 2000)题解与实验指导　姚普选　ISBN 978-7-302-06966-9
数值方法与计算机实现　徐士良　ISBN 978-7-302-11604-2
网络基础及 Internet 实用技术　姚永翘　ISBN 978-7-302-06488-6
网络基础与 Internet 应用　姚永翘　ISBN 978-7-302-13601-9
网络数据库技术与应用　何薇　ISBN 978-7-302-11759-9
网页设计创意与编程　魏善沛　ISBN 978-7-302-12415-3
网页设计创意与编程实验指导　魏善沛　ISBN 978-7-302-14711-4
网页设计与制作技术教程(第 2 版)　王传华　ISBN 978-7-302-15254-8
网页设计与制作教程　杨选辉　ISBN 978-7-302-10686-9
网页设计与制作实验指导　杨选辉　ISBN 978-7-302-10687-6
微型计算机原理与接口技术(第 2 版)　冯博琴　ISBN 978-7-302-15213-2
微型计算机原理与接口技术题解及实验指导(第 2 版)　吴宁　ISBN 978-7-302-16016-8
现代微型计算机原理与接口技术教程　杨文显　ISBN 978-7-302-12761-1
新编 16/32 位微型计算机原理及应用教学指导与习题详解　李继灿　ISBN 978-7-302-13396-4

读者意见反馈

亲爱的读者：

感谢您一直以来对清华版计算机教材的支持和爱护。为了今后为您提供更优秀的教材，请您抽出宝贵的时间来填写下面的意见反馈表，以便我们更好地对本教材做进一步改进。同时如果您在使用本教材的过程中遇到了什么问题，或者有什么好的建议，也请您来信告诉我们。

地址：北京市海淀区双清路学研大厦 A 座 602　　计算机与信息分社营销室 收
邮编：100084　　　　电子邮件：jsjjc@tup.tsinghua.edu.cn
电话：010-62770175-4608/4409　　邮购电话：010-62786544

教材名称：　Web 技术应用基础（第 2 版）

ISBN：978-7-302-18840-7

个人资料

姓名：________ 年龄：____ 所在院校/专业：__________

文化程度：______ 通信地址：____________________

联系电话：______ 电子信箱：____________________

您使用本书是作为：□指定教材 □选用教材 □辅导教材 □自学教材

您对本书封面设计的满意度：

□很满意 □满意 □一般 □不满意　改进建议__________________

您对本书印刷质量的满意度：

□很满意 □满意 □一般 □不满意　改进建议__________________

您对本书的总体满意度：

从语言质量角度看 □很满意 □满意 □一般 □不满意

从科技含量角度看 □很满意 □满意 □一般 □不满意

本书最令您满意的是：

□指导明确 □内容充实 □讲解详尽 □实例丰富

您认为本书在哪些地方应进行修改？（可附页）

____

____

您希望本书在哪些方面进行改进？（可附页）

____

____

电子教案支持

敬爱的教师：

为了配合本课程的教学需要，本教材配有配套的电子教案（素材），有需求的教师可以与我们联系，我们将向使用本教材进行教学的教师免费赠送电子教案（素材），希望有助于教学活动的开展。相关信息请拨打电话 010-62776969 或发送电子邮件至 jsjjc@tup.tsinghua.edu.cn 咨询，也可以到清华大学出版社主页（http://www.tup.com.cn 或 http://www.tup.tsinghua.edu.cn）上查询。